(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 300 454**

(51) Int. Cl.:
G06F 9/445 (2006.01)
H04Q 7/32 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Número de solicitud europea: **02749158 .8**
(86) Fecha de presentación : **22.07.2002**
(87) Número de publicación de la solicitud: **1410189**
(87) Fecha de publicación de la solicitud: **21.04.2004**

(54) Título: **Sistema y método para organizar un software para un dispositivo de comunicación inalámbrica actualizable sobre el terreno.**

(30) Prioridad: **26.07.2001 US 917026**
26.07.2001 US 916460
26.07.2001 US 916900
10.08.2001 US 927131
02.10.2001 US 969305

(45) Fecha de publicación de la mención BOPI:
16.06.2008

(45) Fecha de la publicación del folleto de la patente:
16.06.2008

(73) Titular/es: **Kyocera Wireless Corp.**
10300 Campus Point Drive
San Diego, California 92121, US

(72) Inventor/es: **Rajaram, Gowri;**
Seckendorf, Paul y
Kaplan, Diego

(74) Agente: **Isern Jara, Jorge**

DESCRIPCIÓN

Sistema y método para organizar un software para un dispositivo de comunicación inalámbrica actualizable sobre el terreno.

Antecedentes del invento

Ámbito del invento

El presente invento hace referencia en general a dispositivos de comunicación inalámbricos y, más en particular, a un sistema y un método para organizar un software de forma que se puedan realizar actualizaciones y cambios sobre el terreno en el software de un sistema de un dispositivo de comunicación inalámbrica sobre el terreno a través de una interfaz aérea.

Descripción de la técnica relacionada

No es nada raro que se realicen actualizaciones para un software de teléfonos que ya están en el mercado. Estas actualizaciones pueden estar relacionadas con problemas descubiertos en el software una vez se han fabricado y distribuido los teléfonos al público. Algunas actualizaciones pueden implicar el empleo de nuevas características en el teléfono, o de los servicios ofrecidos por el proveedor del servicio. Sin embargo, otras actualizaciones pueden tener que ver con problemas regionales o problemas asociados a determinadas operadoras de telecomunicaciones. Por ejemplo, en algunas regiones, la distribución de la red de las operadoras puede que imponga situaciones de interfaz aérea que provocan que el microteléfono muestre un comportamiento inesperado como, por ejemplo, una búsqueda de canal incorrecta, una terminación de llamada inapropiada, un audio inadecuado o similares.

El planteamiento tradicional para llevar a cabo estas actualizaciones ha sido entregar el dispositivo de comunicación inalámbrica, denominado asimismo en este documento dispositivo inalámbrico, teléfono o microteléfono, en la tienda de la operadora de telecomunicaciones / punto de asistencia técnica más próximo, o al fabricante para procesar los cambios. Los costes que comportan estas actualizaciones son considerables y reducen el resultado neto. Además, son una molestia para el cliente y es probable que le irriten. A menudo, la solución práctica es dar al cliente un teléfono nuevo.

En US 5.771.386 (DE 19.502.728) (“Baumbauer”) se describe un ejemplo de sistema convencional para configurar software en un dispositivo de telecomunicación. Baumbauer hace referencia a un dispositivo de telecomunicación con un software que consta de varias unidades de programa compilables por separado. A fin de reducir el tiempo necesario para fabricar el dispositivo y hacer los cambios en el software, Baumbauer informa de que cada unidad de programa tiene una cabecera que contiene las direcciones (Ap1, Ap2, Ad) que se utilizan para tratar los procedimientos y/o los datos combinados en las unidades de programa. Además, ofrece un catálogo que contiene referencias para tratar las cabeceras de las unidades de programa. El catálogo está disponible para todas las unidades de programa cargadas. Dentro del marco de la fabricación del dispositivo de telecomunicación, las unidades de programa predefinidas no necesitan estar conectadas cuando se implementa el software del dispositivo de telecomunicación. Baumbauer no informa de ningún dispositivo de comunicación inalámbrica actualizable *in situ*.

Sería conveniente que el software del dispositivo de comunicación inalámbrica pudiera actualizarse por poco dinero y sin molestias para el consumidor.

Sería conveniente que el software del dispositivo de comunicación inalámbrica pudiera actualizarse sin que el cliente pierda el uso de sus teléfonos durante un periodo de tiempo significativo.

Sería conveniente que el software del dispositivo de comunicación inalámbrica pudiera actualizarse en un tiempo de servicio técnico mínimo o sin necesidad de enviar el dispositivo a un servicio técnico.

Sería conveniente que el software del dispositivo de comunicación pudiera organizarse de manera que permitiera realizar modificaciones sobre el terreno a través de una interfaz aérea.

Resumen del invento

Las actualizaciones del software del aparato de comunicación inalámbrica ofrecen a los clientes el mejor producto posible y la mejor experiencia de usuario. La retirada de microteléfonos para actualizar el software es un componente caro del negocio. Estas actualizaciones pueden ser necesarias para ofrecer al usuario servicios adicionales o para abordar los problemas descubiertos durante la utilización del teléfono después de su fabricación. El presente invento hace posible la actualización práctica del software del microteléfono *in situ*, a través de una interfaz aérea.

El presente invento ofrece un método para gestionar el software de un dispositivo de comunicación inalámbrica de acuerdo con la reivindicación 1 y un dispositivo de comunicación inalámbrica que cuenta con un software de sistema actualizable de acuerdo con la reivindicación 12. En las reivindicaciones subordinadas 2 a 11 y 13 a 23, respectivamente, se describen formas de realización preferentes del presente invento.

Por consiguiente, se ofrece un método para organizar el software de un sistema actualizable sobre el terreno de un dispositivo de comunicación inalámbrica. El método comprende: hacer un código de software del sistema en una pluralidad de bibliotecas de símbolos, comprendiendo cada biblioteca símbolos con una funcionalidad relacionada; poner la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código de manera que las bibliotecas de símbolos comiencen al inicio de las secciones de código; crear una segunda pluralidad de bloques de memoria direccionados de forma contigua; identificar cada bloque de memoria con una sección de código correspondiente; almacenar las secciones de código en los bloques de memoria identificados, con el inicio de las secciones de código en las direcciones de inicio correspondientes; mantener una tabla de direcciones de sección de código que cruce los identificadores de sección de código con las direcciones de inicio correspondientes; y ejecutar el software del sistema del dispositivo inalámbrico.

Hacer un código de software del sistema en una primera pluralidad de bibliotecas de símbolos comporta hacer un código de acceso a símbolo. Disponer la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código comprende disponer cada símbolo de manera que se desplace desde su respectiva dirección de inicio de sección de código, y poner el código de acceso a símbolo en una primera sección de código. Después, el método comprende además: mantener una tabla de direcciones de desplazamiento de símbolo que cruce los identificadores de símbolo con las direcciones de desplazamiento correspondientes, e identificadores de sección de código correspondientes; y almacenar la dirección de código de acceso a símbolo en una primera ubicación de la memoria.

La tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo y los datos de lectura-escritura de todas las bibliotecas de símbolos, el código de acceso a símbolo que calcula las direcciones de los símbolos en la biblioteca mientras se ejecuta el software del sistema, y la dirección del código de acceso a símbolo se disponen como bibliotecas de símbolos en una sección de código común, por lo general una sección de código gestor de parches. Los datos de lectura-escritura, la tabla de direcciones de código de sección y el código de acceso a símbolo se cargan y se accede a ellos desde la memoria de lectura-escritura de acceso aleatorio.

Las secciones de código tienen el tamaño necesario para acomodar las bibliotecas de símbolos ordenados o son más grandes que las bibliotecas de símbolos ordenados, en el caso de que las bibliotecas de símbolos se actualicen con bibliotecas más grandes. En los bloques de memoria direccionados de forma contigua caben exactamente las secciones de código correspondientes.

A continuación se ofrecen otros detalles del método descrito más arriba y de una estructura de software para un sistema actualizable *in situ* de un dispositivo de comunicación inalámbrica.

Breve descripción de los dibujos

La figura 1 muestra un esquema de un diagrama de bloque de todo el sistema de mantenimiento del software del dispositivo inalámbrico;

La figura 2 muestra un esquema de un diagrama de bloque del sistema de mantenimiento del software, destacando la instalación de conjuntos de instrucciones a través de una interfaz aérea;

La figura 3 muestra un esquema de un diagrama de bloque de la estructura de software del sistema actualizable *in situ* del presente invento para su utilización en el dispositivo de comunicación inalámbrica;

La figura 4 muestra un esquema de un diagrama de bloque de la memoria del dispositivo inalámbrico;

La figura 5 muestra una tabla que representa la tabla de direcciones de la sección de código de la figura 3;

La figura 6 muestra una representación detallada de la biblioteca de símbolos uno de la figura 3, con símbolos;

La figura 7 muestra una tabla que representa la tabla de direcciones de desplazamiento de símbolo de la figura 3;

Las figuras 8a y 8b muestran diagramas de flujo que ilustran el método del presente invento para organizar un software para un sistema actualizable *in situ* de un dispositivo de comunicación inalámbrica.

Descripción detallada de las formas de realización preferentes del invento

Algunas porciones de las descripciones detalladas que siguen se presentan en términos de procedimientos, pasos, bloques lógicos, códigos, procesamiento y otras representaciones simbólicas de operaciones en bits de datos dentro de un microprocesador o memoria de un dispositivo inalámbrico. Estas descripciones y representaciones son los medios que utilizan los especialistas en las técnicas del tratamiento de datos para comunicar con más eficacia la sustancia de su trabajo a otras personas especializadas en la técnica. Un procedimiento, paso ejecutado por el microprocesador, aplicación, bloque lógico, proceso, etc., se concibe aquí, y en general, para que sea una secuencia de pasos o instrucciones con coherencia propia que se traduzca en un resultado deseado. Los pasos son aquellos que requieren manipulaciones físicas de cantidades físicas. Por lo general, aunque no necesariamente, estas cantidades toman la forma de señales eléctricas o magnéticas que se pueden almacenar, transferir, combinar, comparar y manipular de otro modo en un dispositivo inalámbrico basado en un microprocesador. Se ha demostrado que a veces es útil, sobre todo por razones

de uso común, referirse a estas señales como bits, valores, elementos, símbolos, caracteres, términos, números o equivalentes. Cuando se mencionan dispositivos físicos, como por ejemplo una memoria, éstos están conectados a otros dispositivos físicos por medio de un bus u otra conexión eléctrica. Se puede considerar que estos dispositivos físicos interaccionan con procesos o aplicaciones lógicas y, por lo tanto, se "conectan" a operaciones lógicas. Por ejemplo, una memoria puede almacenar un código de acceso para favorecer una operación lógica.

Habría que tener en cuenta, sin embargo, que todos estos términos y otros similares se asociarán a las cantidades físicas adecuadas y no son más que etiquetas útiles que se aplican a esas cantidades. Salvo que se indique concretamente de otro modo por resultar evidente de los análisis siguientes, se entenderá que en todo el presente invento, los comentarios que utilizan términos tales como "procesar" o "conectar" o "traducir" o "presentar" o "inducir" o "determinar" o "visualizar" o "reconocer" o similares se refieren a la acción y los procesos de un sistema de microprocesador de un dispositivo inalámbrico que manipula y transforma los datos representados como cantidades (electrónicas) físicas en los registros y memorias del sistema informático en otros datos que se representan de manera similar como cantidades físicas en las memorias o registros del dispositivo inalámbrico o en otros dispositivos de almacenamiento, transmisión o presentación de información.

La figura 1 muestra un esquema de un diagrama de bloque de todo el sistema de mantenimiento del software de un dispositivo inalámbrico 100. La organización del software del sistema del presente invento se presenta en detalle más abajo, a continuación de una visión general del sistema de mantenimiento del software 100. El sistema general 100 describe un proceso de suministro de actualizaciones y conjuntos de instrucciones (programas) del software del sistema, y de instalación del software suministrado en un dispositivo inalámbrico. Las actualizaciones del software del sistema o conjuntos de instrucciones de creación de parches (CICP) las crea el fabricante de los microteléfonos. El software del sistema se organiza en bibliotecas de símbolos. Las bibliotecas de símbolos se disponen en secciones de código. Cuando hay que actualizar las bibliotecas de símbolos, la actualización del software 102 se transporta como una o más secciones de código. La actualización del software se transmite a dispositivos inalámbricos *in situ*, de los que el dispositivo de comunicación inalámbrica 104 es representativo, o se transmite en comunicaciones independientes desde una estación base 106 utilizando protocolos de transporte aéreo de datos o mensajes convencionales muy conocidos. El invento no se limita a ningún formato de transporte concreto ya que el dispositivo de comunicación inalámbrica se puede modificar con facilidad para que procese cualquier protocolo de transporte sin hilos disponible para recibir el software del sistema y las actualizaciones de CICP.

El software del sistema se puede ver como una colección de subsistemas diferentes. Los códigos objetos se acoplan firmemente a uno de estos subsistemas abstractos y la colección resultante se etiqueta como biblioteca de símbolos. Esto ofrece un fallo lógico del código base y los parches y arreglos del software se asocian a una de estas bibliotecas de símbolos. En la mayoría de los casos, una sola actualización se asocia a una, o a lo sumo dos, bibliotecas de símbolos. El resto de los códigos base, las demás bibliotecas de símbolos, permanecen sin cambios.

La noción de bibliotecas de símbolos ofrece un mecanismo que permite ocuparse del código y las constantes. Los datos de lectura-escritura (LE), por otra parte, entran en una única biblioteca de LE específica que contiene datos basados en la RAM para todas las bibliotecas.

Una vez recibido por el dispositivo inalámbrico 104, hay que procesar la sección de código transportada. Este dispositivo inalámbrico sobrescribe una sección de código concreta de memoria no volátil 108. La memoria no volátil 108 cuenta con una sección de sistema de archivos (SSA) 110 y una sección de almacenamiento de códigos 112. La sección de código suele comprimirse antes del transporte para minimizar el espacio que la ocupa en la SSA 110. Muchas veces, la sección de código actualizada irá acompañada de datos de LE, que es otro tipo de biblioteca de símbolos que contiene todos los datos de LE de cada biblioteca de símbolos. Aunque se carga en una memoria de lectura-escritura de acceso aleatorio 114 cuando el software del sistema se está ejecutando, es necesario guardar siempre los datos de LE en la memoria no volátil 108 para que se puedan cargar en una memoria de lectura-escritura de acceso aleatorio 114 cada vez que se reinicializar el dispositivo inalámbrico. Esto incluye la primera vez que se cargan los datos de LE en la memoria de lectura-escritura de acceso aleatorio. Como se explica con más detalle a continuación, los datos de LE se disponen por lo general con una sección de código gestor de parches.

El sistema 100 comprende el concepto de tablas virtuales. Con estas tablas se puede arreglar (sustituir) bibliotecas de símbolos de una sección de código, sin romper (sustituir) otras partes del software del sistema (otras secciones de código). A efectos de eficiencia, las tablas virtuales se ejecutan de la memoria de lectura-escritura de acceso aleatorio 114. Una tabla de direcciones de sección de código y una tabla de desplazamiento de símbolo son tablas virtuales.

El dispositivo inalámbrico 104 recibe las secciones de código actualizadas y las almacena en la SSA 110. Una interfaz de usuario (IU) notificará por regla general al usuario que el nuevo software está disponible. En respuesta a las peticiones del IU, el usuario acusa recibo de la notificación y señala el arreglo u operación de actualización. Otra posibilidad es realizar la operación de actualización automáticamente. Es posible que el dispositivo inalámbrico no pueda llevar a cabo las tareas de comunicación estándar mientras se realiza el proceso de actualización. La sección de código gestor de parches tiene una biblioteca de símbolos con un controlador de lectura-escritura no volátil que también se carga en la memoria de lectura-escritura de acceso aleatorio 114. La biblioteca de símbolos con controlador de lectura-escritura no volátil hace que las secciones de código actualizadas se sobrescriban en las secciones de código anteriores. Como se muestra en la figura, la sección de código *n* y las secciones de código gestor de parches están sobrescritas con las secciones de código actualizadas. La sección de código gestor de parches comprende los datos

de lectura-escritura, una tabla de direcciones de sección de código, y una tabla de direcciones de desplazamiento de símbolo, además de un código de acceso a símbolo y la dirección del código de acceso a símbolo (analizados a continuación). Cuando se introducen secciones de código actualizadas, se invalidan partes de estos datos; y una sección de código gestor de parches actualizada consta de datos de lectura-escritura, una tabla de direcciones de sección de código y una tabla de desplazamiento de símbolo válida para las secciones de código actualizadas. Una vez se han cargado las secciones de código actualizadas en la sección de almacenamiento de códigos 112, se reinicia el dispositivo inalámbrico. Tras la operación de reiniciación, el dispositivo inalámbrico puede ejecutar el software del sistema actualizado. Debe entenderse que la sección de código gestor de parches puede constar de otras bibliotecas de símbolos que no se han tratado más arriba. No es necesario cargar estas otras bibliotecas de símbolos en la memoria de lectura-escritura 114.

La figura 2 muestra un esquema de un diagrama de bloque del sistema de mantenimiento del software 100, en la que se destaca la instalación de conjuntos de instrucciones a través de la interfaz aérea. Además de actualizar las secciones de código del software del sistema, el sistema de mantenimiento 100 puede descargar e instalar conjuntos de instrucciones o programas, denominados aquí conjunto de instrucciones del creador de parches (CICP). La sección de código CICP 200 se transporta al dispositivo inalámbrico 104 de la misma forma que las secciones de código del software del sistema descritas anteriormente. Las secciones de código CICP se almacenan en principio en la SSA 110. Una sección de código CICP suele ser un archivo binario que se visualiza en el microteléfono en forma de instrucciones compiladas. Una sección de código CICP es lo bastante extensa como para facilitar la ejecución de operaciones matemáticas básicas y de operaciones ejecutadas con reservas. Por ejemplo, un CICP para calibrar una RF podría llevar a cabo las operaciones siguientes:

SI EL ELEMENTO CAL DE RF ES MENOR QUE X

EJECUTAR INSTRUCCIÓN

SI NO

EJECUTAR INSTRUCCIÓN

Un CICP acepta operaciones matemáticas básicas tales como suma, resta, multiplicación y división. Al igual que las secciones de código del software del sistema, la sección de código CICP se puede cargar en respuesta a las peticiones del IU, y tras cargar el CICP en la sección de almacenamiento de códigos 112, hay que reiniciar el dispositivo inalámbrico. Seguidamente, se ejecuta la sección CICP. Si la sección de código CICP se asocia a cualquier tabla virtual o datos de lectura-escritura, se transportará junto con el CICP una sección de código gestor de parches actualizada, que se instalará en la sección de almacenamiento de códigos 112. Si no, se puede guardar y procesar el CICP del SSA 110. Después de que el microteléfono 104 haya ejecutado todas las instrucciones de la sección CICP, esta sección se puede borrar de la SSA 110.

En algunos aspectos del invento, la organización del software del sistema en bibliotecas de símbolos puede afectar al tamaño de la memoria volátil 114 y de la memoria no volátil 108 necesarias para su ejecución. Ello se debe a que las secciones de código son por lo general más grandes que las bibliotecas de símbolos dispuestas en las secciones de código. Estas secciones de código mayores sirven para alojar las secciones de código actualizadas. Organizar el software del sistema en forma de colección de bibliotecas afecta a los requisitos de tamaño de la memoria no volátil. Para el mismo tamaño de código, la cantidad de memoria no volátil utilizada será mayor debido a que las secciones de código se pueden hacer de un tamaño mayor que el de las bibliotecas de símbolos dispuestas en su interior.

El CICP es un motor de instrucción de tiempo de ejecución muy potente. El microteléfono puede ejecutar cualquier instrucción que se le suministre a través del entorno del CICP Este mecanismo se utiliza para apoyar los calibrados de RF y las actualizaciones de la interfaz de acceso primario. Más en general, el CICP se puede utilizar para depurar a distancia el software del dispositivo inalámbrico cuando el fabricante o el proveedor de servicios reconoce problemas en el software, normalmente a raíz de las quejas del usuario. El CICP también puede registrar los datos necesarios para diagnosticar los problemas del software. El CICP pone en servicio las aplicaciones del sistema recién descargadas con el fin de analizar datos, realizar depuraciones y arreglos. El CICP puede facilitar actualizaciones basadas en los datos de LE para analizar y posiblemente arreglar a corto plazo un problema en vez de una sección de código de software del sistema actualizada. El CICP ofrece algoritmos para comprimir la memoria que se utilizan con el dispositivo inalámbrico.

Una vez se han suministrado las actualizaciones del software al dispositivo inalámbrico, el sistema de mantenimiento del software 100 soporta la compresión de memoria. La compresión de memoria es parecida a las aplicaciones de desfragmentación del disco de los ordenadores de sobremesa. El mecanismo de compresión garantiza que la memoria se utiliza de forma óptima y está bien equilibrada para futuras actualizaciones de sección de código en que el tamaño de las secciones de código actualizadas son impredecibles. El sistema 100 analiza la sección de almacenamiento de códigos mientras se está parcheando (actualizando). El sistema 100 intenta introducir las secciones de código actualizadas en el espacio de memoria ocupado por la sección de código que se está sustituyendo. Si la sección de código actualizada es más grande que la sección de código que se está sustituyendo, el sistema 100 comprime las secciones de código en la memoria 112. Otra posibilidad es que el fabricante o el proveedor de servicios calcule la compresión, y entonces las instrucciones de compresión se transportan al dispositivo inalámbrico 104.

La compresión puede ser un proceso muy largo debido a la complejidad del algoritmo y también al vasto volumen de movimiento de datos. El algoritmo de compresión predice la viabilidad antes de empezar cualquier proceso. Las peticiones del IU se utilizan para solicitar permiso al usuario antes de intentar llevar a cabo la compresión.

En algunos aspectos del invento, todas las secciones de código del software del sistema se actualizan al mismo tiempo. Sin embargo, una actualización completa del software del sistema requeriría una SSA 110 más grande.

La figura 3 muestra un esquema de un diagrama de bloque del presente invento de estructura de software de sistema actualizable sobre el terreno para su uso en el dispositivo de comunicación inalámbrica 104. La estructura de software del sistema 300 comprende un software de sistema ejecutable diferenciado en una segunda pluralidad de secciones de código. Se muestran las secciones de código uno (302) dos (304) y *n* (306), así como de la sección de código gestor de parches 308; sin embargo, el invento no se limita a un número concreto de secciones de código. El sistema 300 comprende además una primera pluralidad de bibliotecas de símbolos dispuestas en la segunda pluralidad de secciones de código. Se muestran la biblioteca de símbolos uno (310) en la sección de código uno (302), las bibliotecas de símbolos dos (312) y tres (314) dispuestas en la sección de código dos (304), y la biblioteca de símbolos *n* (316), dispuesta en la sección de código *n* (306). Cada biblioteca consta de símbolos que tienen una funcionalidad afín. Por ejemplo, la biblioteca de símbolos uno (310) puede participar en el funcionamiento de la pantalla de cristal líquido (LCD) del dispositivo inalámbrico. Después, los símbolos se asociarían a funciones de visualización. Como se explica en detalle más abajo, las bibliotecas de símbolos adicionales están en la sección de código gestor de parches 308.

La figura 4 muestra un esquema de un diagrama de bloque de la memoria del dispositivo inalámbrico. Tal como se muestra, la memoria es la sección de almacenamiento de códigos 112 de la figura 1. La memoria es una memoria no volátil en la que se puede escribir, como por ejemplo una memoria Flash. Debe entenderse que las secciones de código no tienen que almacenarse necesariamente en la misma memoria que la SSA 110. Debe entenderse asimismo que la estructura de software del sistema del presente invento podría activarse con secciones de código almacenadas en una pluralidad de memorias cooperadoras. La sección de almacenamiento de códigos 112 incluye una segunda pluralidad de bloques de memoria direccionados de forma contigua, donde cada bloque de memoria almacena una sección de código correspondiente de la segunda pluralidad de secciones de código. Así, la sección de código uno (302) se guarda en un primer bloque de memoria 400, la sección de código dos (304) en el segundo bloque de memoria 402, la sección de código *n* (306) en el bloque de memoria n^{avo} 404, y la sección de código gestor de parches (308) en el bloque de memoria p^{avo} 406.

Si se comparan las figuras 3 y 4, el inicio de cada sección de código se almacena en las direcciones de inicio correspondientes de la memoria, y las bibliotecas de símbolos están dispuestas para empezar al inicio de las secciones de código. Es decir, cada biblioteca de símbolos empieza en una primera dirección y pasa por una serie de direcciones en secuencia desde la primera dirección. Por ejemplo, la sección de código uno (302) empieza en la primera dirección de inicio 408 (señalada con una "S") de la memoria de la sección de almacenamiento de códigos 112. En la figura 3, la biblioteca de símbolos uno (310) empieza al principio 318 de la primera sección de código. Del mismo modo, la sección de código dos (304) empieza en una segunda dirección de inicio 410 (figura 4), y la biblioteca de símbolos dos empieza al principio 320 de la sección de código dos (figura 3). La sección de código *n* (306) empieza en una tercera dirección de inicio 412 de la memoria de la sección de almacenamiento de códigos 112 (figura 4), y la biblioteca de símbolos *n* (316) empieza en el principio de la sección de códigos *n* 322 (figura 3). La sección de código gestor de parche empieza en la dirección de inicio p^{avo} 414 de la sección de almacenamiento de códigos 112, y la primera biblioteca de símbolos de la sección de código gestor de parches 308 empieza al principio 324 de la sección de código gestor de parches. Así, la biblioteca de símbolos uno (310) se almacena finalmente en el primer bloque de memoria 400. Si una sección de código tiene una pluralidad de bibliotecas de símbolos, como la sección de código dos (304), la pluralidad de bibliotecas de símbolos se guarda en el bloque de memoria correspondiente, en este caso el segundo bloque de memoria 402.

En la figura 3, la estructura de software del sistema 300 comprende además una tabla de direcciones de sección de código 326 en forma de un tipo de símbolo incluido en una biblioteca de símbolos dispuesta en la sección de código gestor de parches 308. La tabla de direcciones de la sección de código cruza en la memoria los identificadores de la sección de código con las direcciones de inicio de sección de código correspondientes.

La figura 5 muestra una tabla que representa la tabla de direcciones de sección de código 326 de la figura 3. Para encontrar la dirección de inicio de la sección de código de una biblioteca de símbolos, se consulta la tabla de direcciones de sección de código 326. Por ejemplo, el sistema 300 busca la sección de código uno cuando se necesita un símbolo de la biblioteca de símbolos uno para ejecutarlo. Para encontrar la dirección de inicio de la sección de código uno, y localizar, por lo tanto, el símbolo en la biblioteca de símbolos uno, se consulta la tabla de direcciones de sección de código 326. La disposición de las bibliotecas de símbolos en secciones de código y la localización de las secciones de código con una tabla permite mover o ampliar las secciones de código. Para instalar secciones de código actualizadas (con bibliotecas de símbolos actualizadas), puede que sea necesario llevar a cabo las operaciones de ampliación o de desplazamiento.

Volviendo a la figura 3, debe tenerse en cuenta que no todas las bibliotecas de símbolos empiezan necesariamente al principio de una sección de código. Como se muestra, la biblioteca de símbolos tres (314) se encuentra en la sección de código dos (304), pero no empieza la dirección de inicio de la sección de código 320. De modo que, si se necesita

un símbolo de una biblioteca de símbolos tres (314) para ejecutarlo, el sistema 300 consulta la tabla de direcciones de sección de código 326 para buscar la dirección de inicio de la sección de código dos (304). Como se explica más abajo, una tabla de direcciones de desplazamiento de símbolo permite localizar los símbolos en la biblioteca de símbolos tres (314). No importa que los símbolos estén esparcidos por múltiples bibliotecas mientras se mantengan en la misma sección de código.

Como se ha observado anteriormente, cada biblioteca de símbolos contiene símbolos relacionados funcionalmente. Un símbolo es un nombre definido por el programador para localizar y utilizar un cuerpo de rutina, una variable, o una estructura de datos. Así, un símbolo puede ser una dirección o un valor. Los símbolos pueden ser internos o externos. Los símbolos internos no se pueden ver fuera del ámbito de la sección de código actual. Más en concreto, otras bibliotecas de símbolos no los buscan en otras secciones de código. Los símbolos externos se utilizan y se solicitan por las secciones de código y las bibliotecas los buscan en diferentes secciones de código. La tabla de direcciones de desplazamiento de símbolo suelen incluir una lista de todos los símbolos externos.

Por ejemplo, la biblioteca de símbolos uno puede generar caracteres en una pantalla de un dispositivo inalámbrico. Los símbolos de esta biblioteca generarían, a su vez, números de teléfono, nombres, la hora u otras características de pantalla. Cada característica se genera con rutinas, denominadas aquí símbolos. Por ejemplo, una símbolo de la biblioteca de símbolos uno (310) genera números de teléfono en la pantalla. Este símbolo se representa con una "X" y es externo. Cuando el dispositivo inalámbrico recibe una llamada telefónica y el servicio de ID de la persona que llama está activado, el sistema debe ejecutar el símbolo "X" para generar el número en la pantalla. Por lo tanto, el sistema debe localizar el símbolo "X".

La figura 6 muestra una representación detallada de la biblioteca de símbolos uno (310) de la figura 3, con símbolos. Los símbolos están contenidos para que se desplacen de las respectivas direcciones de inicio de la sección de código. En muchas circunstancias, el principio de la biblioteca de símbolos es el principio de una sección de código, pero esto no es así si una sección de código tiene más de una biblioteca de símbolos. La biblioteca de símbolos uno empieza al principio de la sección de código uno (véase la figura 3). Como se muestra en la figura 6, el símbolo "X" se encuentra en un desplazamiento transversal (03) del principio de la biblioteca de símbolos y el símbolo "Y" se encuentra en un desplazamiento transversal (15). Las direcciones de desplazamiento de símbolo se guardan en una tabla de direcciones de desplazamiento de símbolo 328 de la sección de código gestor de parches (véase la figura 3).

La figura 7 muestra una tabla que representa la tabla de direcciones de desplazamiento de símbolo 328 de la figura 3. La tabla de direcciones de desplazamiento de símbolo 328 cruza en la memoria los identificadores de símbolo con las direcciones de desplazamiento correspondientes y con los identificadores de sección de código correspondientes. Así, cuando el sistema trata de ejecutar el símbolo "X" de la biblioteca de símbolos uno, consulta la tabla de direcciones de desplazamiento de símbolo 328 para localizar la dirección exacta del símbolo con respecto a la sección de código en que está situada.

Volviendo a la figura 3, la primera pluralidad de bibliotecas de símbolos incluye por lo general todos los datos de lectura-escritura que deben consultarse o fijar en la ejecución de estas bibliotecas de símbolos. Por ejemplo, una biblioteca de símbolos puede comprender una operación que depende de una instrucción condicional. La sección de datos de lectura-escritura se consulta para determinar el estatus necesario para completar la instrucción condicional. El presente invento agrupa los datos de lectura-escritura de todas las bibliotecas de símbolos en una sección de lectura-escritura compartida. En algunos aspectos del invento, los datos de lectura-escritura 330 están en la sección de código gestor de parches 308. Otra posibilidad (no se muestra) es disponer los datos de lectura-escritura en una sección de código diferente, por ejemplo, en la sección de código *n* (306).

La primera pluralidad de bibliotecas de símbolos incluye asimismo un código de acceso a símbolo situado en una sección de código para calcular la dirección del símbolo solicitado. El código de acceso a símbolo se encuentra y se almacena en una dirección de una sección de código aparte, la sección de código 2 (304), por ejemplo. Sin embargo, tal como se muestra, el código de acceso a símbolo 332 se encuentra y se almacena en una dirección de la sección de código gestor de parches 308. La estructura de software del sistema 300 comprende además una primera ubicación para almacenar la dirección del código de acceso a símbolo. La primera ubicación se encuentra en una sección de código de la sección de almacenamiento de códigos 112, o en una sección de memoria aparte del dispositivo inalámbrico (no se muestra). La primera ubicación también se puede disponer en la misma sección de código que los datos de lectura-escritura. Como se muestra, la primera ubicación 334 se almacena en la sección de código gestor de parches 308 con los datos de lectura-escritura 330, la tabla de direcciones de desplazamiento de símbolo, la tabla de direcciones de la sección de código 326 y el código de acceso a símbolo 332 y la biblioteca de parches (biblioteca de símbolos de parche) 336.

El código de acceso a símbolo utiliza la tabla de direcciones de sección de código y las tablas de direcciones de desplazamiento de símbolo para buscar en la memoria la dirección exacta del símbolo solicitado. O sea, el código de acceso a símbolo accede a la tabla de direcciones de sección de código y a la tabla de direcciones de desplazamiento de símbolo para calcular la dirección del símbolo que busca. Por ejemplo, si se busca el símbolo "X" en la biblioteca de símbolos uno, se solicita el acceso a símbolo para buscar el identificador del símbolo (ID de símbolo) X_1, correspondiente al símbolo "X" (véase la figura 7). El código de acceso a símbolo consulta la dirección de desplazamiento de símbolo para establecer que el identificador de símbolo X_1 tiene un desplazamiento transversal de (03) a partir del inicio de la sección de código uno (véase la figura 6). El código de acceso a símbolo se solicita para buscar el

identificador de la sección de código CS_1, correspondiente a la sección de código uno. El código de acceso a símbolo consulta la tabla de direcciones de sección de código para establecer la dirección de inicio asociada al identificador de la sección de código (ID de sección de código) CS_1. De este modo, el código de acceso a símbolo determina que el identificador de símbolo X_1 está desplazado (03) de la dirección (00100) o se encuentra en la dirección (00103).

El símbolo "X" es un nombre reservado porque forma parte del código real. En otras palabras, tiene unos datos absolutos asociados al mismo. Los datos pueden ser una dirección o un valor. El identificador de símbolo es un alias creado para seguir al símbolo. La tabla de direcciones de desplazamiento de símbolo y la tabla de direcciones de sección de código funcionan con identificadores para evitar la confusión con los nombres de símbolo y de sección de código reservados. También es posible que se utilice el mismo nombre de símbolo en muchas bibliotecas de símbolos. La utilización de identificadores impide la confusión entre estos símbolos.

Volviendo a la figura 1, la estructura de software del sistema 100 comprende además una memoria de lectura-escritura 114, generalmente una memoria de acceso aleatorio (RAM). Los datos de lectura-escritura 330, la tabla de direcciones de sección de código 326, la tabla de direcciones de desplazamiento de símbolo 328, el código de acceso a símbolo 332 y la dirección del código de acceso a símbolo 334 se cargan en la memoria de lectura-escritura 114 desde la sección gestor de parches para poder acceder durante la ejecución del software del sistema. Como se sabe, los tiempos de acceso para un código almacenado en la RAM son significativamente inferiores a los de acceso a una memoria no volátil como la Flash.

Volviendo a la figura 3, se observa que las bibliotecas de símbolos no necesariamente llenan las secciones de código en las que se encuentran, aunque los bloques de memoria tengan el tamaño necesario para alojar exactamente las secciones de código correspondientes almacenadas en su interior. Otra posibilidad indicada es que cada segunda pluralidad de secciones de código tenga un tamaño en bytes que permita alojar las bibliotecas de símbolos dispuestas, y que cada bloque de memoria direccionado de forma contigua tenga un tamaño en bytes que permita alojar las secciones de código correspondientes. Por ejemplo, la sección de código uno (302) puede ser una sección de 100 bytes para alojar una biblioteca de símbolos que tenga una longitud de 100 bytes. El primer bloque de memoria sería de 100 bytes para que coincida con el tamaño de bytes de la sección de código uno. Sin embargo, la biblioteca de símbolos cargada en la sección de código uno puede tener menos de 100 bytes. Como se muestra en la figura 3, la sección de código uno (302) tiene una sección no utilizada 340 ya que la biblioteca de símbolos uno (310) tiene menos de 100 bytes. Así, cada segunda pluralidad de las secciones de código puede tener un tamaño mayor que el necesario para alojar las bibliotecas de símbolos dispuestas en su interior. "Sobredimensionar" las secciones de código permite alojar bibliotecas de símbolos actualizadas más grandes.

Las figuras 8a y 8b muestran diagramas de flujo que ilustran el método del presente invento para organizar el software de un sistema actualizable sobre el terreno de un dispositivo de comunicación inalámbrica. Aunque para que el método se vea más claro se describe como una secuencia de pasos numerados, de la numeración no deberá inferirse ningún orden, a no ser que se indique explícitamente. El método empieza en el paso 800. El paso 802 forma el software de sistema en una primera pluralidad de bibliotecas de símbolos; cada biblioteca de símbolos comprende símbolos que tienen una funcionalidad relacionada. El paso 804 dispone la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código. El paso 806 ejecuta el software del sistema del dispositivo inalámbrico.

Disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código incluye iniciar las bibliotecas de símbolos al principio de las secciones de código, y el método comprende otros pasos. El paso 805a almacena el inicio de las secciones de código en las direcciones de inicio correspondientes. El paso 805b mantiene una tabla de direcciones de sección de código cruzando los identificadores de sección de código con las direcciones de inicio correspondientes.

Disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código consiste en disponer los símbolos que se desplazarán de sus respectivas direcciones de inicio de sección de código. Después, el paso 805c mantiene una tabla de direcciones de desplazamiento de símbolo que cruza identificadores de símbolo con las direcciones de desplazamiento correspondientes y con los identificadores de sección correspondientes.

En algunos aspectos del invento, formar en el paso 802 un código de software de sistema en una primera pluralidad de bibliotecas de símbolos consiste en crear datos de lectura-escritura para la pluralidad de bibliotecas de símbolos. Disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código incluye disponer los datos de lectura-escritura en una sección de código de lectura-escritura compartida.

En algunos aspectos del invento, formar en el paso 802 un código de software de sistema en una primera pluralidad de bibliotecas de símbolos comprende crear un código de acceso a símbolo, y disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código comprende disponer el código de acceso a símbolo en una primera sección de código. Luego, en el paso 806, ejecutar las secciones de código como software de sistema incluye varios subpasos. El paso 806a carga una tercera pluralidad de bibliotecas de símbolos en una memoria de lectura-escritura, generalmente la RAM. La tercera pluralidad de bibliotecas de símbolos no necesariamente tiene que contener todas las bibliotecas de símbolos de la sección de código gestor de parches. La tercera pluralidad de bibliotecas de símbolos contiene asimismo bibliotecas de símbolos que están en otras secciones de código, aparte de la sección de código gestor de parches. El paso 806b, en respuesta a la referencia a la primera

ubicación de la memoria, accede al código de acceso a símbolo. El paso 806c pide al código de acceso a símbolo que calcule la dirección del símbolo solicitado utilizando el identificador de símbolo correspondiente y un identificador de sección de código correspondiente. El paso 806d accede a la tercera pluralidad de bibliotecas de símbolos de la RAM.

Solicitar, en el paso 806b, el código de acceso a símbolo para calcular la dirección del símbolo solicitado comprende acceder a la tabla de direcciones de sección de código y a la tabla de direcciones de desplazamiento de símbolo para calcular la dirección de dicho símbolo. Por lo general, almacenar, en el paso 805d, la dirección del código de acceso a símbolo en una primera ubicación de la memoria comprende almacenar la dirección del código de acceso a símbolo en la primera sección de código.

En algunos aspectos del invento, disponer en el paso 804 la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código consiste en poner los datos de lectura-escritura, la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo, y el código de acceso a símbolo en la primera sección de código, por lo general la sección de código gestor de parches. Después, el paso 806a, carga los datos de lectura-escritura, la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo, el código de acceso a símbolo, y la dirección de código de acceso a símbolo de la primera sección de código de la memoria de lectura-escritura (generalmente, la RAM). El paso 806d accede a los datos de lectura-escritura, la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo, el código de acceso a símbolo, y la dirección de código de acceso a símbolo de la memoria de lectura-escritura.

Almacenar en el paso 805a el inicio de las secciones de código en las direcciones de inicio correspondientes consta de varios subpasos. El paso 805a1 crea una segunda pluralidad de bloques de memoria direccionados de forma contigua. El paso 805a2 identifica cada bloque de memoria con una sección de código correspondiente. El paso 805a3 guarda las secciones de código en los bloques de memoria identificados.

En algunos aspectos del invento, disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código consiste en poner una tercera pluralidad de bibliotecas de símbolos en una primera sección de código. Después, en el paso 805a2, identificar cada bloque de memoria con una sección de código correspondiente comprende identificar un primer bloque de memoria con la primera sección de código, y almacenar, en el paso 805a3, secciones de código en los bloques de memoria identificados incluye almacenar la tercera pluralidad de bibliotecas de símbolos en el primer bloque de memoria.

Como alternativa, disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código comprende poner una primera biblioteca de símbolos en una primera sección de código. Después, en el paso 805a3, identificar cada bloque de memoria con una sección de código correspondiente incluye identificar un primer bloque de memoria con la primera sección de código, y almacenar, en el paso 805a3, las secciones de código en los bloques de memoria identificados incluye almacenar la primera biblioteca de símbolos en el primer bloque de memoria.

Disponer, en el paso 804, la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código comprende establecer el tamaño de las secciones de código para que puedan alojar las bibliotecas de símbolos dispuestas en las mismas. Después, en el paso 805a1, crear una segunda pluralidad de bloques de memoria direccionados de forma contigua comprende establecer el tamaño de los bloques de memoria para que puedan alojar las secciones de código correspondientes. Si no, en el paso 804, disponer la primera pluralidad de bibliotecas de símbolos en una segunda pluralidad de secciones de código comprende establecer el tamaño de las secciones de código para que puedan acomodar tamaños más grandes que las bibliotecas de símbolos dispuestas en las mismas.

Se ha ofrecido un sistema y método para organizar un software de sistema de un dispositivo inalámbrico para realizar actualizaciones *in situ* a través de una interfaz aérea. El sistema se puede actualizar con facilidad gracias a la disposición de bibliotecas de símbolos en secciones de código, con tablas para acceder a las direcciones de inicio de las secciones de código de la memoria y a las direcciones de desplazamiento de símbolo de las bibliotecas de símbolos. Aunque se han dado unos cuantos ejemplos de estas disposiciones de las bibliotecas y tablas de referencia cruzada para una función de pantalla, el presente invento no se limita simplemente a estos ejemplos. A los especialistas en la técnica se les ocurrirán otras variantes y formas de realizaciones del invento.

REIVINDICACIONES

1. Método para gestionar software en un dispositivo de comunicación inalámbrica (104), comprendiendo dicho método: disponer una pluralidad de símbolos en una pluralidad de secciones de código (302, 304, 306, 308); crear una tabla de direcciones de desplazamiento de símbolo (328) que cruce los símbolos con las direcciones de desplazamiento correspondientes y las secciones de código correspondientes (302, 304, 306, 308); recibir una actualización del software (102) a través de una comunicación inalámbrica, comprendiendo la actualización del software (102) un símbolo actualizado y una porción de tabla de direcciones de desplazamiento de símbolo actualizada para actualizar al menos una parte de la tabla de direcciones de desplazamiento de símbolo (328); almacenar el símbolo actualizado en una de las secciones de código (302, 304, 306, 308); incorporar la porción de tabla de direcciones de desplazamiento de símbolo actualizada en la tabla de direcciones de desplazamiento de símbolo (328); almacenar el inicio (318, 320, 322, 324, 408, 410, 412, 414) de las secciones de código (302, 304, 306, 308) en las direcciones de inicio correspondientes; y mantener una tabla de direcciones de sección de código que cruce las secciones de código (302, 304, 306, 308) con las direcciones de inicio correspondientes (318, 320, 322, 324, 408, 410, 412, 414).

2. Método de la reivindicación 1, en el que un símbolo comprende datos de lectura-escritura (330) y en el que el paso de colocación comprende además disponer los datos de lectura-escritura (330) en una sección de código de lectura-escritura compartida.

3. Método de la reivindicación 1, en el que el paso de creación comprende además: crear un código de acceso a símbolo (332); disponer el código de acceso a símbolo (332) en una sección de código (302, 304, 306, 308); almacenar la dirección del código de acceso a símbolo (332) en la memoria; y pedir el código de acceso a símbolo (332) para calcular la dirección de un símbolo solicitado utilizando un identificador de símbolo correspondiente y un identificador de sección de código correspondiente.

4. Método de la reivindicación 3, en el que el paso de petición comprende además acceder a la tabla de direcciones de sección de código y a la tabla de direcciones de desplazamiento de símbolo (328) para calcular la dirección del símbolo solicitado.

5. Método de la reivindicación 3, en el que almacenar la dirección del código de acceso a símbolo (334) en una primera ubicación de la memoria comprende almacenar la dirección del código de acceso a símbolo (334) en la primera sección de código (302, 304, 306, 308).

6. Método de la reivindicación 3, en el que el paso de disposición del código de acceso a símbolo (332) comprende además poner los datos de lectura-escritura (330), la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo (328) y el código de acceso a símbolo (332) en una sola de las secciones de código (302, 304, 306, 308).

7. Método de la reivindicación 6, comprendiendo además el paso de ejecutar un software de sistema actualizable *in situ*, incluyendo el paso de ejecución: cargar los datos de lectura-escritura (330), la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo (328), el código de acceso a símbolo (332) y la dirección del código de acceso a símbolo (332) de una sección de código (302, 304, 306, 308) en una memoria de lectura-escritura (114); y acceder a los datos de lectura-escritura (330), la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo (328), el código de acceso a símbolo (332) y la dirección del código de acceso a símbolo (334) de la memoria de lectura-escritura.

8. Método de la reivindicación 1, en el que almacenar el inicio (318, 320, 322, 324, 408, 410, 412, 414) de las secciones de código (302, 304, 306, 308) en las direcciones de inicio correspondientes (318, 320, 322, 324, 408, 410, 412, 414) incluye: crear una pluralidad de bloques de memoria direccionados de forma contigua; identificar cada bloque de memoria con una sección de código correspondiente (302, 304, 306, 308); y almacenar las secciones de código (302, 304, 306, 308) en los respectivos bloques de memoria identificados.

9. Método de la reivindicación 8, en el que disponer la pluralidad de símbolos en una pluralidad de secciones de código (302, 304, 306, 308) comprende poner dos o más símbolos en una sola sección de código (302, 304, 306, 308).

10. Método de la reivindicación 8, en el que disponer la pluralidad de símbolos en una pluralidad de secciones de código (302, 304, 306, 308) comprende poner un único símbolo en una sola sección de código (302, 304, 306, 308).

11. Método de la reivindicación 8, en el que el paso de disponer la primera pluralidad de símbolos comprende establecer el tamaño de una sección de código (302, 304, 306, 308) para que pueda alojar tamaños más grandes que el símbolo correspondiente.

12. Dispositivo de comunicación inalámbrica (104) con un software de sistema actualizable *in situ*, comprendiendo: un software de sistema ejecutable, actualizable *in situ*, diferenciado en una pluralidad de secciones de código (302, 304, 306, 308), en el que cada sección de código (302, 304, 306, 308) comprende uno o más símbolos, teniendo los símbolos las direcciones de desplazamiento correspondientes, comprendiendo desplazamientos desde las direcciones de inicio (318, 320, 322, 324, 408, 410, 412, 414) de las secciones de código correspondientes (302, 304, 306, 308); una

tabla de direcciones de desplazamiento de símbolo (328) que cruza símbolos con las direcciones de desplazamiento correspondientes y las secciones de código correspondientes (302, 304, 306, 308); y una tabla de direcciones de sección de código que cruza las secciones de código (302, 304, 306, 308) con las direcciones de inicio de sección de código correspondientes (318, 320, 322, 324, 408, 410, 412, 414) en la memoria.

13. Dispositivo de comunicación inalámbrica de la reivindicación 12, en el que la pluralidad de secciones de código (302, 304, 306, 308) comprende una sección de lectura-escritura compartida, y en el que la pluralidad de bibliotecas de símbolos comprende datos de lectura-escritura (330) que se encuentran en la sección de lectura-escritura.

14. Dispositivo de comunicación inalámbrica de la reivindicación 12, en el que la pluralidad de símbolos comprende un código de acceso a símbolo (332) que está en una primera sección de código (302, 304, 306, 308) para calcular la dirección de un símbolo solicitado, y en el que la memoria comprende el código de acceso a símbolo (332) almacenado en una dirección, comprendiendo además: una primera ubicación para almacenar la dirección del código de acceso a símbolo (334).

15. Dispositivo de comunicación inalámbrica de la reivindicación 14, en el que el código de acceso a símbolo (332) accede a la tabla de direcciones de la sección código y a la tabla de direcciones de desplazamiento de símbolo (328) para calcular la dirección del símbolo solicitado.

16. Dispositivo de comunicación inalámbrica de la reivindicación 15, en el que la primera ubicación está en la primera sección de código (302, 304, 306, 308).

17. Dispositivo de comunicación inalámbrica de la reivindicación 15, en el que uno o más símbolos comprenden datos de lectura-escritura (330), la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo (328), y el código de acceso a símbolo (332) que están en la primera sección de código (302, 304, 306, 308).

18. Dispositivo de comunicación inalámbrica de la reivindicación 17, comprendiendo además: una memoria volátil de lectura-escritura (114), en la que los datos de lectura-escritura (330), la tabla de direcciones de sección de código, la tabla de direcciones de desplazamiento de símbolo (328), el código de acceso a símbolo (332) y la dirección del código de acceso a símbolo (334) se cargan en la memoria de lectura-escritura (114) de la primera sección de código (302, 304, 306, 308) para acceder a ellos durante la ejecución del software del sistema.

19. Dispositivo de comunicación inalámbrica de la reivindicación 12, en el que la memoria comprende una pluralidad de bloques de memoria direccionados de forma contigua, almacenando cada bloque de memoria una sección de código correspondiente (302, 304, 306, 308) de la pluralidad de secciones de código (302, 304, 306, 308).

20. Dispositivo de comunicación inalámbrica de la reivindicación 19, comprendiendo además un bloque de memoria para almacenar una primera sección de código (302, 304, 306, 308), en el que la primera sección de código (302, 304, 306, 308) comprende una pluralidad de bibliotecas de símbolos que están en la primera sección de código (302, 304, 306, 308).

21. Dispositivo de comunicación inalámbrica de la reivindicación 19, comprendiendo además un bloque de memoria para almacenar una primera sección de código (302, 304, 306, 308), en el que la primera sección de código (302, 304, 306, 308) comprende una sola biblioteca de símbolos dispuesta en la primera sección de código (302, 304, 306, 308).

22. Dispositivo de comunicación inalámbrica de la reivindicación 19, en el que el tamaño de cada pluralidad de secciones de código (302, 304, 306, 308) es mayor que el tamaño necesario para alojar los símbolos contenidos en la misma.

23. Dispositivo de comunicación inalámbrica de la reivindicación 12, en el que la memoria es una memoria no volátil en la que se puede escribir.

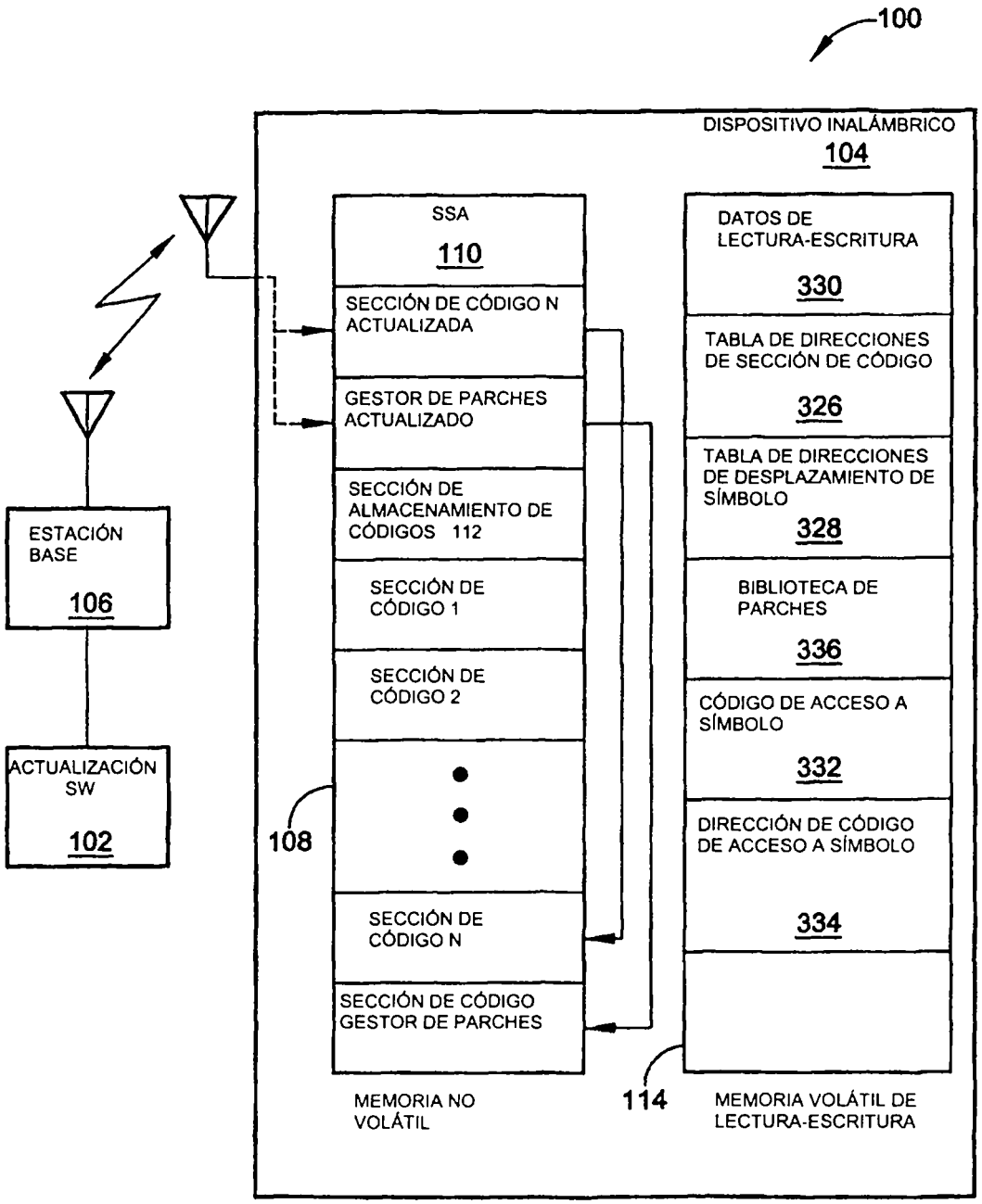
100
DISPOSITIVO INALÁMBRICO
104
SSA
110
SECCIÓN DE CÓDIGO N ACTUALIZADA
GESTOR DE PARCHES ACTUALIZADO
SECCIÓN DE ALMACENAMIENTO DE CÓDIGOS 112
SECCIÓN DE CÓDIGO 1
SECCIÓN DE CÓDIGO 2
SECCIÓN DE CÓDIGO N
SECCIÓN DE CÓDIGO GESTOR DE PARCHES
108
MEMORIA NO VOLÁTIL
DATOS DE LECTURA-ESCRITURA
330
TABLA DE DIRECCIONES DE SECCIÓN DE CÓDIGO
326
TABLA DE DIRECCIONES DE DESPLAZAMIENTO DE SÍMBOLO
328
BIBLIOTECA DE PARCHES
336
CÓDIGO DE ACCESO A SÍMBOLO
332
DIRECCIÓN DE CÓDIGO DE ACCESO A SÍMBOLO
334
114
MEMORIA VOLÁTIL DE LECTURA-ESCRITURA
ESTACIÓN BASE
106
ACTUALIZACIÓN SW
102

FIG. 1

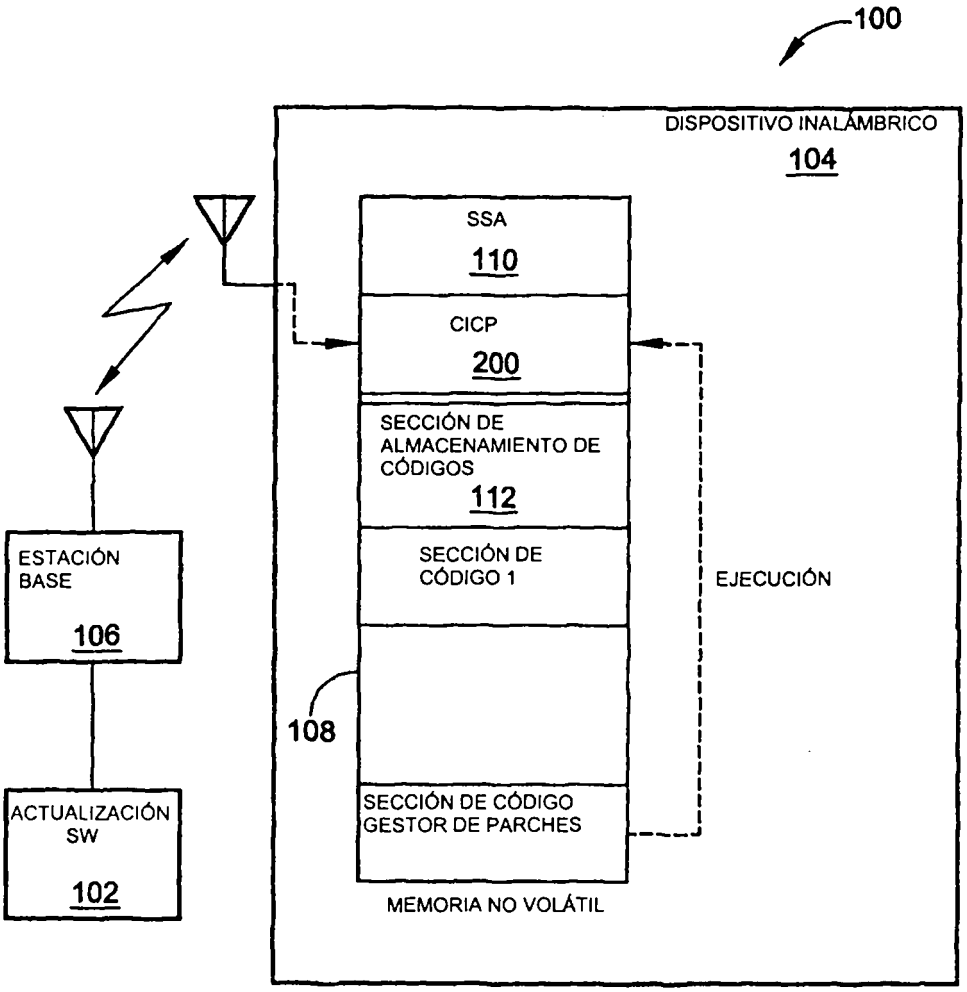
100
DISPOSITIVO INALÁMBRICO
104
SSA
110
CICP
200
SECCIÓN DE ALMACENAMIENTO DE CÓDIGOS
112
SECCIÓN DE CÓDIGO 1
EJECUCIÓN
108
SECCIÓN DE CÓDIGO GESTOR DE PARCHES
MEMORIA NO VOLÁTIL
ESTACIÓN BASE
106
ACTUALIZACIÓN SW
102

FIG. 2

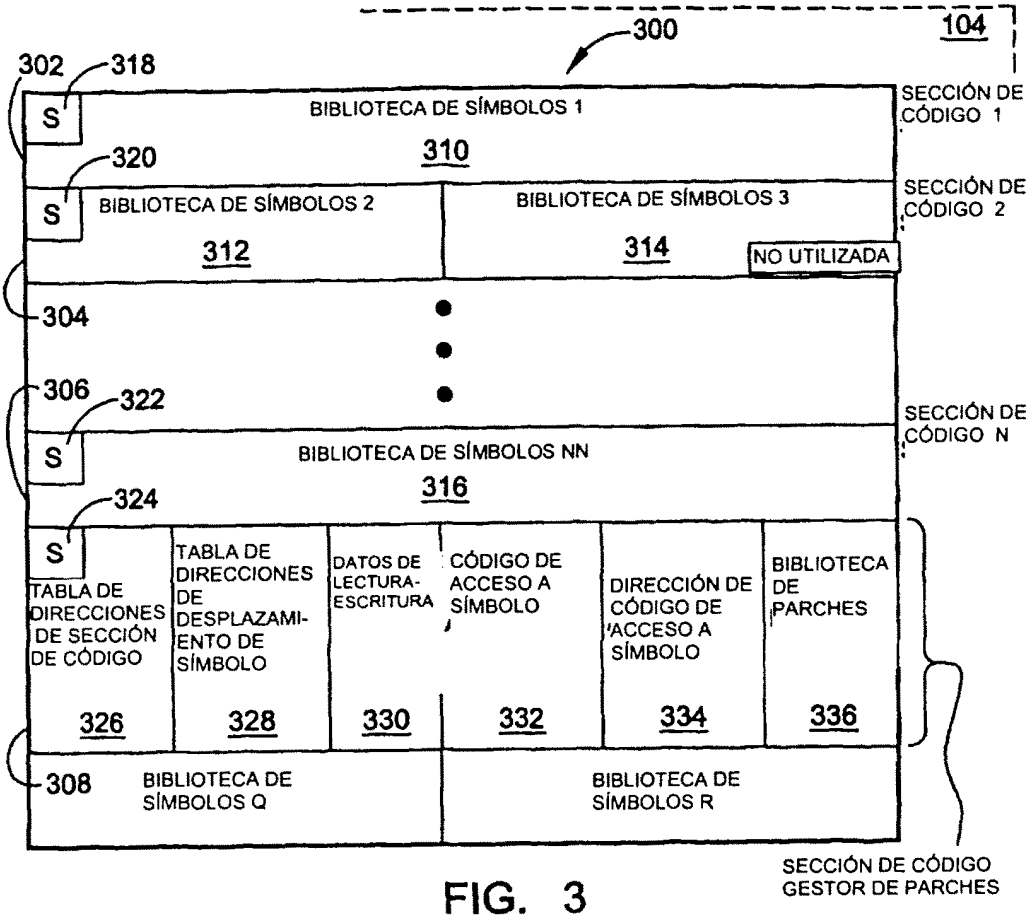
300
104
302
318
S
BIBLIOTECA DE SÍMBOLOS 1
310
SECCIÓN DE CÓDIGO 1
320
S
BIBLIOTECA DE SÍMBOLOS 2
312
BIBLIOTECA DE SÍMBOLOS 3
314
NO UTILIZADA
SECCIÓN DE CÓDIGO 2
304
306
322
S
BIBLIOTECA DE SÍMBOLOS NN
316
SECCIÓN DE CÓDIGO N
324
S
TABLA DE DIRECCIONES DE SECCIÓN DE CÓDIGO
326
TABLA DE DIRECCIONES DE DESPLAZAMIENTO DE SÍMBOLO
328
DATOS DE LECTURA-ESCRITURA
330
CÓDIGO DE ACCESO A SÍMBOLO
332
DIRECCIÓN DE CÓDIGO DE ACCESO A SÍMBOLO
334
BIBLIOTECA DE PARCHES
336
308
BIBLIOTECA DE SÍMBOLOS Q
BIBLIOTECA DE SÍMBOLOS R
SECCIÓN DE CÓDIGO GESTOR DE PARCHES

FIG. 3

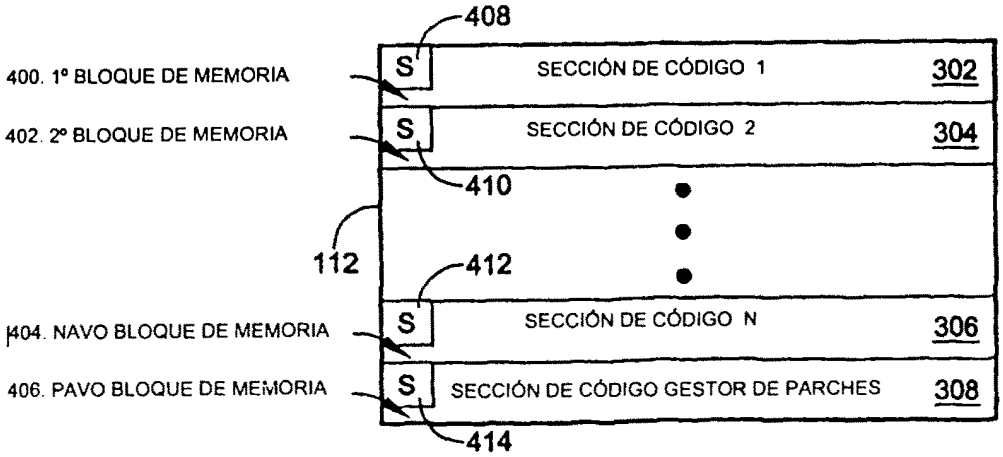
408
400. 1º BLOQUE DE MEMORIA
S
SECCIÓN DE CÓDIGO 1
302
402. 2º BLOQUE DE MEMORIA
S
SECCIÓN DE CÓDIGO 2
304
410
112
412
404. NAVO BLOQUE DE MEMORIA
S
SECCIÓN DE CÓDIGO N
306
406. PAVO BLOQUE DE MEMORIA
S
SECCIÓN DE CÓDIGO GESTOR DE PARCHES
308
414

FIG. 4

326

TABLA DE DIRECCIONES DE SECCIÓN DE CÓDIGO	
IDENTIFICADOR DE SECCIÓN DE CÓDIGO	DIRECCIÓN
CS _ 1	DIRECCIÓN DE INICIO 1 (00100)
CS _ 2	DIRECCIÓN DE INICIO 2 (00200)
● ● ●	
CS _ N	DIRECCIÓN DE INICIO N (00700)
PM	DIRECCIÓN DE INICIO P (01000)

FIG. 5

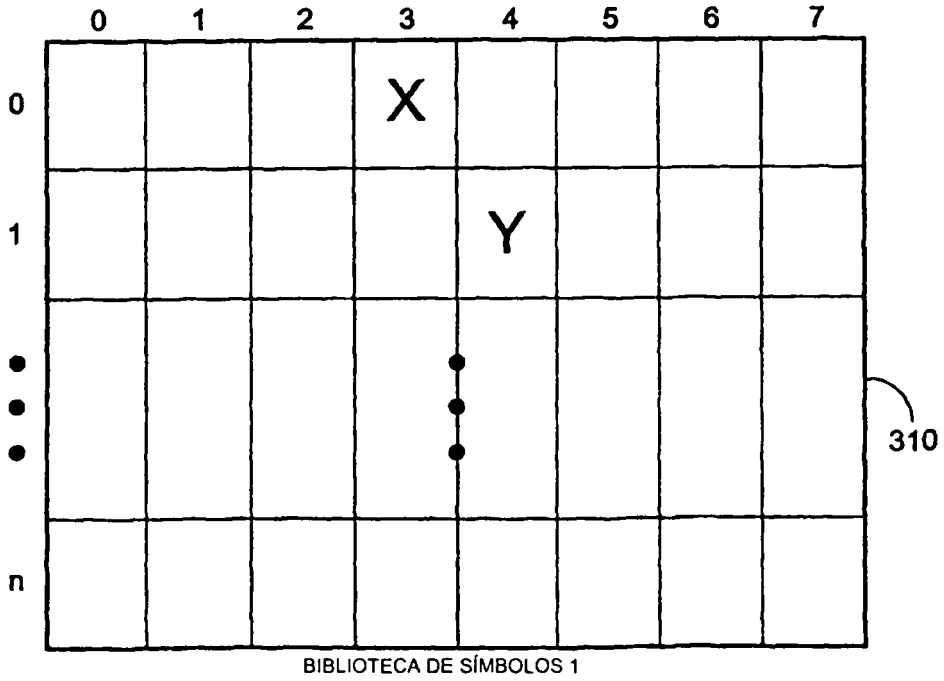

FIG. 6

TABLA DE DIRECCIONES DE DESPLAZAMIENTO DE SÍMBOLO		
ID SÍMBOLO	ID SECCIÓN DE CÓDIGO	DESPLAZAMIENTO
X_1	CS_1	03
Y_1	CS_1	15
P_1	CS_2	11
Q_1	CS_2	33
AA_3	CS_2	47
• • •		

FIG. 7

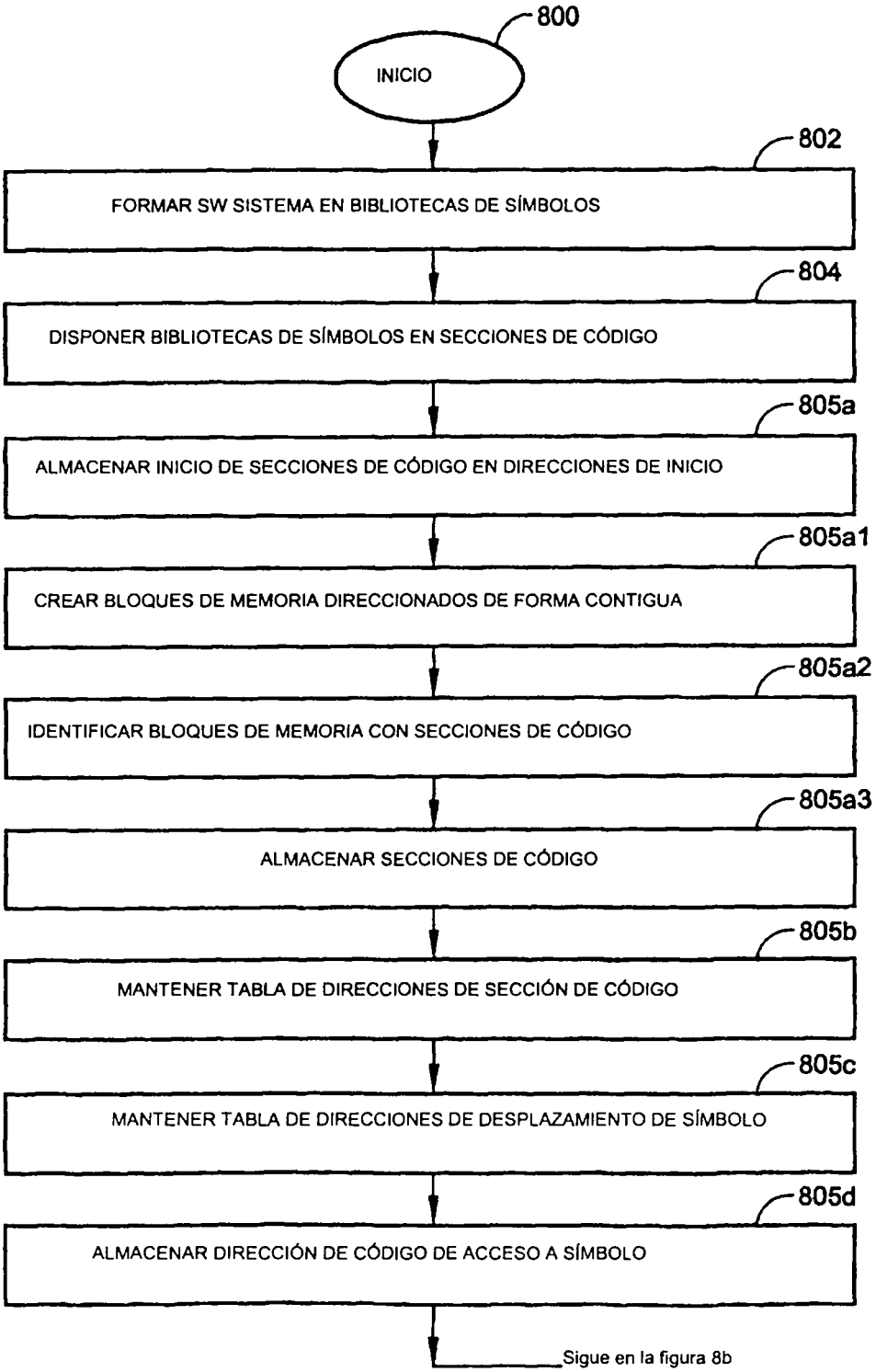
800
INICIO
802
FORMAR SW SISTEMA EN BIBLIOTECAS DE SÍMBOLOS
804
DISPONER BIBLIOTECAS DE SÍMBOLOS EN SECCIONES DE CÓDIGO
805a
ALMACENAR INICIO DE SECCIONES DE CÓDIGO EN DIRECCIONES DE INICIO
805a1
CREAR BLOQUES DE MEMORIA DIRECCIONADOS DE FORMA CONTIGUA
805a2
IDENTIFICAR BLOQUES DE MEMORIA CON SECCIONES DE CÓDIGO
805a3
ALMACENAR SECCIONES DE CÓDIGO
805b
MANTENER TABLA DE DIRECCIONES DE SECCIÓN DE CÓDIGO
805c
MANTENER TABLA DE DIRECCIONES DE DESPLAZAMIENTO DE SÍMBOLO
805d
ALMACENAR DIRECCIÓN DE CÓDIGO DE ACCESO A SÍMBOLO
Sigue en la figura 8b

FIG. 8a

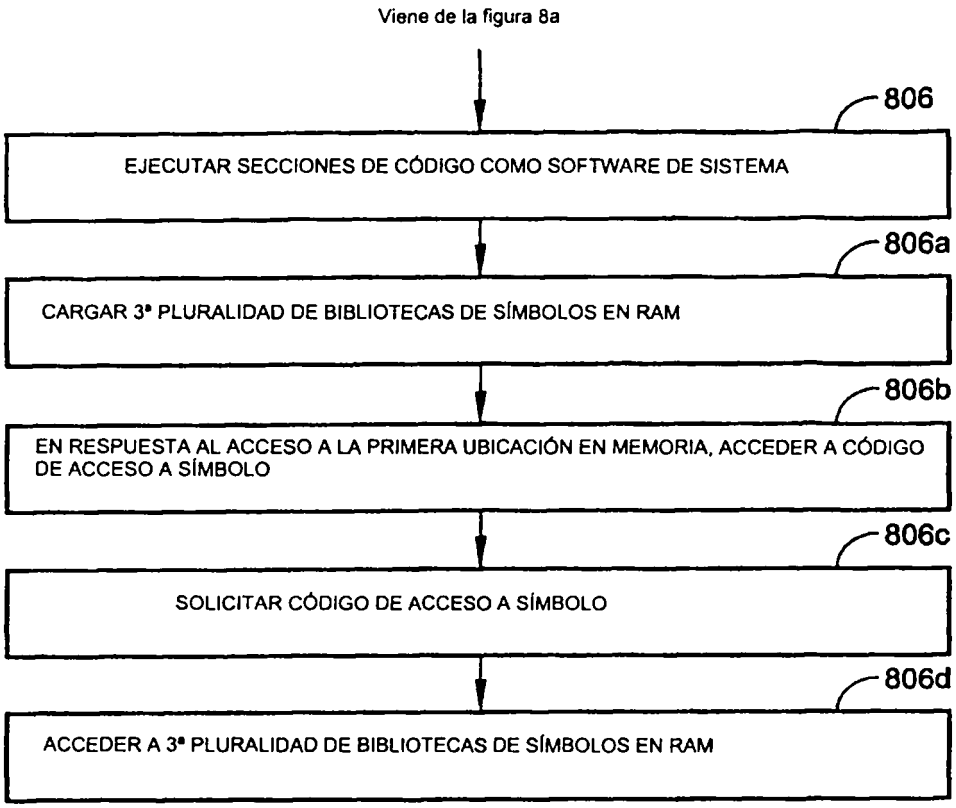
Viene de la figura 8a
806
EJECUTAR SECCIONES DE CÓDIGO COMO SOFTWARE DE SISTEMA
806a
CARGAR 3ª PLURALIDAD DE BIBLIOTECAS DE SÍMBOLOS EN RAM
806b
EN RESPUESTA AL ACCESO A LA PRIMERA UBICACIÓN EN MEMORIA, ACCEDER A CÓDIGO DE ACCESO A SÍMBOLO
806c
SOLICITAR CÓDIGO DE ACCESO A SÍMBOLO
806d
ACCEDER A 3ª PLURALIDAD DE BIBLIOTECAS DE SÍMBOLOS EN RAM

FIG. 8b