(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 805 460**

(51) Int. CI.:

G06F 11/07 (2006.01)
G06F 11/30 (2006.01)
G06F 11/34 (2006.01)
G06F 8/61 (2008.01)
G06F 9/445 (2008.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **24.09.2018 E 18196330 (7)**

(97) Fecha y número de publicación de la concesión europea: **17.06.2020 EP 3460664**

(54) Título: **Procedimiento de gestión de módulos de software integrados para una computadora electrónica de un dispositivo de conmutación eléctrica**

(30) Prioridad:

25.09.2017 FR 1758826

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
12.02.2021

(73) Titular/es:

SCHNEIDER ELECTRIC INDUSTRIES SAS (100.0%)
35, rue Joseph Monier
92500 Rueil-Malmaison, FR

(72) Inventor/es:

BUFFERNE, VINCENT y
POPOVIC, VLADIMIR

(74) Agente/Representante:

GONZÁLEZ PECES, Gustavo Adolfo

DESCRIPCIÓN

Procedimiento de gestión de módulos de software integrados para una computadora electrónica de un dispositivo de conmutación eléctrica

La presente invención se refiere a un procedimiento de gestión de módulos de software integrados ejecutados por una computadora electrónica integrada en un dispositivo de conmutación eléctrica. La invención también se refiere a un dispositivo de conmutación eléctrica que comprende una computadora electrónica integrada programada para la implementación de este procedimiento de gestión.

La invención se refiere en particular al campo de los dispositivos de conmutación eléctrica, como los disyuntores de bajo voltaje, provistos de una computadora electrónica integrada para ejecutar el software integrado.

Hoy en día, este tipo de software permite la implementación de funcionalidades avanzadas que mejoran el funcionamiento de estos dispositivos, como por ejemplo, funciones de control remoto, supervisión y gestión.

Las solicitudes de patentes US 2009/072022 A1 y US 2010/031243 A1 describen ejemplos de este tipo de software integrado en dispositivos de conmutación.

Es deseable poder evolucionar estos programas de software con el tiempo y de acuerdo a las necesidades de los usuarios. Por ejemplo, los usuarios pueden optar por utilizar únicamente las funcionalidades que les interesan en función del uso previsto del dispositivo. También pueden querer añadir o eliminar ciertas funciones si sus necesidades cambian con el tiempo. Esto también permite al fabricante actualizar los dispositivos a lo largo de su vida útil, o incluso ofrecer a los usuarios funcionalidades específicas adaptadas a necesidades particulares. En efecto, no siempre es posible saber, al diseñar un producto, qué uso le darán los usuarios a lo largo de su ciclo de vida.

Tal comportamiento evolutivo debe ser fácil y rápido de implementar por los usuarios.

Sin embargo, las funciones implementadas por este software pueden estar relacionadas con funciones esenciales de los dispositivos eléctricos, como las funciones de protección y de medición eléctrica, por ejemplo para activar un disyuntor en caso de un fallo eléctrico. Por lo tanto, estos programas de software deben cumplir con las restricciones de funcionamiento en tiempo real cuando se ejecutan. De lo contrario, la seguridad operativa del dispositivo se vería comprometida, lo cual es indeseable.

Es bien sabido que algunos conmutadores ofrecen software integrado en tiempo real en el que se pueden instalar varias funcionalidades de forma modular. Pero esta modularidad sigue siendo limitada y tiene un aspecto "monolítico", en el sentido de que las diferentes funcionalidades son realizadas por tareas que se agrupan. En otras palabras, estas funcionalidades son interdependientes, desde la instalación hasta la ejecución. El usuario no puede instalar o desinstalar funciones individuales, sino sólo conjuntos fijos de funcionalidades predefinidas. Así pues, el funcionamiento de ese sistema sigue siendo complicado para el usuario. Esta arquitectura también plantea un problema cuando el fabricante quiere probar y certificar diferentes funcionalidades antes de ofrecerlas a los usuarios. Esto se debe a que las tareas son inseparables dentro del mismo conjunto, y es este conjunto de tareas el que debe ser probado. Si el fabricante desea probar o certificar todas las combinaciones posibles de funcionalidades a fin de ofrecer a los usuarios más posibilidades de elección, entonces el número de conjuntos a probar y certificar aumenta exponencialmente, por lo que esa tarea no es factible en la práctica.

Hay otros sistemas de software con una arquitectura modular, que permiten un uso más flexible. Sin embargo, no garantizan un comportamiento en tiempo real. Por lo tanto, su uso no es deseable.

La invención busca más particularmente remediar estos inconvenientes proponiendo un dispositivo de conmutación de la corriente eléctrica que comprende una computadora electrónica integrada y un procedimiento de funcionamiento asociado, en el que el software integrado tiene un diseño modular que es fácil y rápido de implementar por los usuarios, sin comprometer la seguridad operacional del dispositivo.

Con este fin, la invención se refiere a un procedimiento de gestión de módulos de software integrados para una computadora electrónica integrada de un dispositivo eléctrico de conmutación de una corriente eléctrica, este procedimiento comprende los pasos:

a) la adquisición, mediante un sistema de gestión de módulos de software de la computadora electrónica del dispositivo de conmutación, de un módulo de software que comprende un código ejecutable y un contrato de servicios en el que se declaren los recursos de hardware que requiere el código ejecutable al ser ejecutado por la computadora electrónica ;

b) la instalación del módulo de software dentro de un receptáculo de acogida por parte del sistema de gestión, perteneciendo el receptáculo de acogida a un conjunto de receptáculos de acogida, cada uno de los cuales está destinado a formar un entorno para ejecutar un módulo de software, comprendiendo cada receptáculo para este fin una ubicación de memoria definida estáticamente dentro de una memoria de la computadora electrónica y estando asociado con un subconjunto de recursos de hardware de la computadora electrónica;

c) la ejecución del módulo de software por la computadora electrónica; en el paso c) se incluye, al ejecutar el módulo de software

un paso que consiste en comprobar, mediante el sistema de gestión, si la ejecución del módulo de software cumple con el contrato de servicio para dicho módulo de software, se autoriza la continuación de la ejecución si se cumple el contrato de servicio y, si se identifica la ejecución del módulo de software como no conforme al contrato de servicio, se implementa un paso de recuperación con el fin de interrumpir la ejecución del módulo de software por la computadora electrónica y en el paso c), la ejecución del código comprende la ejecución de varias tareas en tiempo real por una unidad lógica de la computadora electrónica, estando cada tarea asociada a una frecuencia de ejecución predefinida, y la ejecución de las tareas por la unidad lógica de cálculo se regula mediante un primer planificador implementado por el sistema de gestión y luego por un segundo planificador implementado por un sistema operativo en tiempo real de la computadora electrónica, el primer planificador autoriza selectivamente la ejecución de las tareas pendientes de ejecución en función de su frecuencia de ejecución y les asigna un nivel de prioridad de ejecución para el segundo planificador que depende de la frecuencia de ejecución definida para esta tarea.

De esta manera, la invención permite conciliar tanto el aspecto modular del software incorporado como el respeto de las restricciones de ejecución en tiempo real de este software.

Como se ha explicado anteriormente, la estructura modular de los módulos de software es posible en parte por los receptáculos. Esta estructura modular también está permitida por el contrato de servicios, así como por los mecanismos de verificación del cumplimiento del contrato de servicios. El contrato de servicios también garantiza el funcionamiento en tiempo real de los módulos de software.

Como los módulos de software tienen una estructura modular, pueden ser instalados o desinstalados dinámicamente. La gestión del software del dispositivo se facilita a los usuarios. La estructura modular también ofrece más posibilidades para seguir desarrollando el funcionamiento del dispositivo a lo largo del tiempo. Además, al preservar el funcionamiento en tiempo real del software incorporado, la seguridad operacional del dispositivo no se ve afectada.

Dependiendo de los aspectos ventajosos pero no obligatorios de la invención, dicho procedimiento puede incorporar una o más de las siguientes características, tomadas aisladamente o en cualquier combinación técnicamente permisible:

- El procedimiento incluye, antes del paso b), un paso preliminar de verificación del contrato de servicio del módulo de software, rechazándose la instalación del módulo de software si los recursos de hardware requeridos por el contrato de servicio no son compatibles con los recursos de hardware ofrecidos por los receptáculos disponibles.

- Paralelamente al paso c), se está ejecutando también al menos un segundo módulo de software instalado en otro receptáculo diferente, y se está implementando también un segundo paso de verificación de si la ejecución de ese segundo módulo de software cumple el contrato de servicios de ese segundo módulo de software.

- Un primer grupo de tareas agrupa las tareas asociadas a una primera frecuencia de ejecución predefinida, un segundo grupo de tareas agrupa las tareas asociadas a una segunda frecuencia de ejecución predefinida inferior a la primera frecuencia de ejecución, y al ejecutar el módulo de software, la ejecución de las tareas pertenecientes al primer grupo es activada por el planificador cada vez que se recibe una señal de temporización periódica y la ejecución de las tareas pertenecientes al segundo grupo es activada periódicamente por el primer planificador, teniendo las tareas pertenecientes al segundo grupo una prioridad inferior a las tareas pertenecientes al primer grupo.

- El dispositivo de conmutación está conectado a una instalación eléctrica para interrumpir una corriente eléctrica alterna dentro de esta instalación eléctrica, seleccionándose las frecuencias de ejecución en función de la frecuencia de la corriente eléctrica alterna.

- La estrategia de recuperación implementada durante el paso de recuperación en caso de incumplimiento del contrato de servicio de un módulo de software incluye detener la ejecución del módulo de software.

- La estrategia de recuperación implementada durante el paso de recuperación en caso de incumplimiento del contrato de servicio de un módulo de software se selecciona en función de un valor de criticidad del código ejecutable declarado en el contrato de servicio de ese módulo de software.

Según otro aspecto, la invención se refiere a un dispositivo de conmutación de una corriente eléctrica que comprende una computadora electrónica integrada capaz de controlar el funcionamiento del dispositivo de conmutación y destinada a recibir módulos de software integrados, comprendiendo la computadora electrónica un conjunto de receptáculos de acogida destinados cada uno a formar un entorno para ejecutar un módulo de software, comprendiendo cada receptáculo para este propósito una ubicación de memoria definida estáticamente dentro de una memoria de la computadora electrónica y que está asociada con un subconjunto de recursos de hardware de la

computadora electrónica, esta computadora electrónica además implementa un sistema de gestión programado para implementar los pasos:

a) la adquisición de un módulo de software que comprende un código ejecutable y un contrato de servicios en el que se declaren los recursos de hardware que requiere el código ejecutable al ser ejecutado por la computadora electrónica;

b) la instalación del módulo de software dentro de uno de los receptáculos;

c) la ejecución del módulo de software por la computadora electrónica, incluyendo el paso c), durante la ejecución del módulo de software, un paso consistente en verificar si la ejecución del módulo de software cumple con el contrato de servicio para dicho módulo de software, se autoriza la continuación de la ejecución si se cumple el contrato de servicio y, si se identifica la ejecución del módulo de software como no conforme al contrato de servicio, se implementa un paso de recuperación con el fin de interrumpir la ejecución del módulo de software por la computadora electrónica

y en el paso c), la ejecución del código comprende la ejecución de una pluralidad de tareas en tiempo real por una unidad lógica de cálculo de la computadora electrónica, asociándose cada tarea a una frecuencia de ejecución predefinida, y la ejecución de las tareas por la unidad lógica de cálculo está regulada por un primer planificador implementado por el sistema de gestión y luego por un segundo planificador implementado por un sistema operativo en tiempo real de la computadora electrónica, autorizando el primer planificador selectivamente la ejecución de las tareas pendientes de ejecución en función de su frecuencia de ejecución y les asigna un nivel de prioridad de ejecución para el segundo planificador que depende de la frecuencia de ejecución definida para esta tarea.

Según los aspectos ventajosos pero no obligatorios de la invención, dicho dispositivo de conmutación puede tener la siguiente característica: la computadora electrónica se programa además para implementar, antes del paso b), un paso preliminar de verificación del contrato de servicio del módulo de software, rechazándose la instalación del módulo de software si los recursos de hardware requeridos por el contrato de servicio no son compatibles con los recursos de hardware ofrecidos por los receptáculos disponibles.

La invención será mejor entendida y otras ventajas de la misma aparecerán más claramente a la luz de la siguiente descripción de un modo de realización de un procedimiento de gestión de módulos de software dentro de un dispositivo de conmutación de una corriente eléctrica, dada únicamente a modo de ejemplo y con referencia a los dibujos adjuntos, en los cuales:

- La figura 1 es una representación esquemática de un dispositivo de conmutación equipado con una computadora electrónica integrada conforme a la invención;
- La figura 2 es un sinóptico de un sistema de gestión de módulos de software implementado por la computadora electrónica de la figura 1;
- La figura 3 es un esquema simplificado de un módulo de software y un receptáculo para recibir este módulo de software e implementado por el sistema de gestión en la figura 2;
- La figura 4 es un sinóptico de una máquina de estados asociada a un módulo de software de la figura 3;
- Las figuras 5 y 6 son diagramas de flujo que ilustran un procedimiento para operar un planificador del sistema de gestión de la figura 2;
- La figura 7 es un diagrama de flujo de un procedimiento para operar el sistema que se muestra en la figura 2;
- La figura 8 es un diagrama de flujo de un procedimiento de verificación de un contrato de servicios asociado a un módulo de software implementado durante el procedimiento de la figura 7 durante la instalación de un módulo de software;
- La figura 9 es un diagrama de flujo de un procedimiento de verificación de un contrato de servicios asociado a un módulo de software implementado durante el procedimiento de la figura 7 durante la ejecución de un módulo de software;
- Las figuras 10 a 13 son diagramas de flujo de un procedimiento para medir el tiempo de ejecución de una tarea implementada por un módulo de software durante su ejecución dentro del sistema 2;
- La figura 14 es un diagrama de flujo de un procedimiento para implementar una estrategia de recuperación de un módulo de software tras un incumplimiento del contrato de servicio de ese software durante su ejecución en el Sistema 2.

La figura 1 muestra un dispositivo de conmutación 2 de una corriente eléctrica, que comprende un bloque de conmutación 4 y los terminales 6 y 8 para la entrada/salida de corriente eléctrica. El dispositivo 2 está destinado a ser conectado eléctricamente a través de los terminales 6 y 8 a una instalación eléctrica, como un circuito de distribución eléctrica.

El bloque de conmutación 4 es adecuado para interrumpir reversiblemente el flujo de corriente eléctrica entre los terminales 6 y 8, por ejemplo, mediante contactos eléctricos separables cuyo desplazamiento es controlado por un actuador controlable.

Las terminales 6 y 8 se muestran aquí esquemáticamente. Su número exacto puede variar, dependiendo de si la corriente eléctrica es monofásica o polifásica. Por ejemplo, el dispositivo 2 es un dispositivo unipolar o dispositivo de cuatro polos.

En este ejemplo, el dispositivo 2 es un disyuntor de bajo voltaje, adecuado para operar con corriente continua o alterna. Alternativamente, puede ser un disyuntor de alto voltaje, o un contactor, o un interruptor, o cualquier otro dispositivo de conmutación equivalente.

El dispositivo 2 también tiene una computadora electrónica 10 integrada, que es particularmente adecuada para controlar el funcionamiento del dispositivo 2, en este caso por medio de software integrado.

Por ejemplo, la computadora 10 se adapta para controlar el bloque de corte 4 actuando sobre el mencionado actuador controlable.

La computadora 10 consiste en una unidad lógica de computación 12, una memoria informática 14 y una interfaz de usuario 16, conectadas entre sí por un bus de datos interno, no mostrado.

La unidad lógica 12 aquí es un microprocesador o un microcontrolador programable.

En este ejemplo, la memoria 14 consiste en la memoria no volátil 141 y la memoria de trabajo 142. Por ejemplo, la memoria 141 incluye un módulo de memoria no volátil, de tipo Flash u otra tecnología equivalente. La memoria de trabajo 142 es aquí de tipo RAM por "Random Access Memory" en lengua inglesa.

La interfaz de usuario 16 está adaptada para permitir al usuario interactuar con la computadora 10 y, en particular, para añadir y quitar módulos de software.

Por ejemplo, la interfaz 16 incluye una pantalla de visualización controlable, en este caso una pantalla de cristal líquido, así como medios de entrada de datos como botones o una pantalla táctil.

La interfaz 16 también puede incluir medios de conexión de datos para establecer un enlace de datos cableado o inalámbrico entre la computadora 10 y el equipo informático externo para descargar módulos de software. Por ejemplo, la Interfaz 16 incluye un conector de tipo USB con cable o un puerto en serie. La interfaz 16 también puede incluir una interfaz de radio de tipo NFC para la ”near field communication”, una de tipo UNB para ”ultra narrow band” en inglés, o cualquier otra tecnología apropiada.

La interfaz 16 también puede incluir una unidad lectora capaz de leer el contenido de un medio de grabación extraíble, como una tarjeta de memoria o un disco óptico, de nuevo para garantizar la descarga de módulos de software.

Aquí, la computadora 10 está alojada junto con el bloque 4 en la misma carcasa del dispositivo 2. La interfaz 16 se coloca preferentemente de manera que un usuario pueda acceder a ella desde fuera de esta carcasa.

La memoria 14 almacena las instrucciones del software ejecutable para asegurar el funcionamiento de la computadora 10 y el dispositivo 2 cuando estas instrucciones son ejecutadas por la unidad lógica 12. Estas instrucciones toman la forma, por ejemplo, de uno o más programas de software integrados.

La computadora 10 está así adaptada para ejecutar, durante su funcionamiento, por medio de instrucciones de software ejecutables, un sistema operativo 18 en tiempo real y un sistema de gestión 20 de módulos de software. Concretamente, en este ejemplo, el sistema de gestión 20 es ejecutado dentro del sistema operativo 18.

Se sabe que la función del sistema operativo 18 es gestionar el acceso de los programas informáticos a los recursos de hardware de la computadora 10 mientras asegura la ejecución de estos programas informáticos en tiempo real.

A los efectos de esta descripción, se dice que un sistema informático es "en tiempo real" cuando incluye medios que permiten garantizar la ejecución de una tarea con un tiempo de ejecución inferior a una duración predefinida.

El sistema operativo 18 tiene aquí una función de separación espacial.

Las figuras 2 a 4 muestran un ejemplo del sistema de gestión 20.

La función del sistema de gestión 20 es permitir que uno o más módulos de software sean ejecutados en tiempo real por la computadora 10, mientras que permite la instalación y desinstalación dinámica de esos módulos de software, como se explica a continuación. Así, estos módulos de software pueden ser cargados o descargados "en caliente" y se dice que son "enchufables".

A continuación, los módulos de software llevan la referencia "40". En aras de la simplicidad, sólo se describe un módulo 40, de forma genérica, aunque en la práctica se pueden utilizar varios módulos 40 diferentes simultáneamente en el sistema 20.

Por ejemplo, cada módulo de software 40 es un archivo informático que puede ser grabado en la memoria 14.

Cada módulo de software 40 contiene aquí las instrucciones ejecutables 42, o código ejecutable, destinadas a implementar una función predefinida del dispositivo 2 cuando son ejecutadas por la unidad lógica 12. En particular, estas funciones se basan aquí en una o más tareas o rutinas elementales predefinidas.

Cada módulo de software 40 incluye también un contrato de servicio 44, que describe el comportamiento dinámico del módulo 40 durante su ejecución y, en particular, define los recursos requeridos por el módulo 40 al ejecutar el código ejecutable 42. Preferiblemente, estos recursos necesarios corresponden a un valor máximo de recursos requeridos en un caso extremo, también llamado "worst case scenario" en inglés.

En particular, el contrato de servicio 44 contiene una declaración del nivel máximo de uso de los recursos de computación de la unidad lógica 12. Esta información permite determinar el tiempo máximo de ejecución necesario para la realización de las tareas correspondientes por la unidad 12.

El contrato de servicio 44 también contiene declaraciones que indican los requisitos del módulo 40 para otros tipos de recursos de hardware de la computadora 10 y/o el sistema 20 y/o el dispositivo 2, en particular cuando se trata de recursos de hardware que están limitados dentro del dispositivo 2 y que deben ser supervisados para evitar conflictos entre los diferentes módulos 40.

Por ejemplo, el contrato de servicio 44 contiene declaraciones sobre el tamaño máximo de la memoria RAM utilizada por el módulo, o el número máximo de parámetros de configuración gestionados por el módulo 40.

Favorablemente, el contrato de servicio 44 también contiene una declaración sobre el nivel de criticidad de las funciones implementadas por el módulo 40.

Por ejemplo, si el módulo 40 implementa una función relacionada con la seguridad del dispositivo 2, como la protección eléctrica, entonces su nivel de criticidad es más alto que el de un módulo 40 que implementa una función de menor importancia, como una función de medición.

Preferiblemente, el contrato de servicio se almacena como un archivo informático en formato de datos en serie legibles por máquina.

Como ejemplo ilustrativo, el contrato de servicio 44 contiene aquí un archivo informático en formato JSON, por "JavaScript Object Notation". Sin embargo, son posibles otros formatos, en particular el formato XML por "Extended Markup Language". Las declaraciones se almacenan en el contrato de servicios 44 en forma de variables informáticas como valores numéricos, valores booleanos y/o cadenas de caracteres alfanuméricos. Alternativamente, el contrato de servicio 44 puede también estar en forma binaria o tener una codificación específica.

Por ejemplo, el contrato de servicio 44 se genera al escribir y desarrollar el código ejecutable 42.

Dentro del sistema 20, la modularidad del software incorporado está asegurada por un conjunto de 50 receptáculos de acogida, cada uno diseñado para recibir uno de los módulos de software 40. Estos 50 receptáculos también se conocen como “emplacement”, o "slot" en inglés.

Los receptáculos 50 son estructuras de datos adaptadas para recibir un módulo de software 40 y formar un entorno de ejecución para este módulo de software 40.

Por ejemplo, cada receptáculo 50 tiene una ubicación de memoria definida estáticamente, también conocida como contenedor, dentro de una zona 54 de la memoria 14. Específicamente, cada receptáculo 50 define una partición de memoria predefinida con un tamaño de memoria predefinido reservado dentro de la zona 54.

Cada receptáculo 50 define también un conjunto de tareas predefinidas, un conjunto de objetos de software para la interacción con otros sistemas de software implementados por el sistema operativo 18 y una biblioteca de funciones que pueden ser utilizadas por el módulo de software 40 alojado en ese receptáculo 50.

Por lo tanto, cada receptáculo 50 está asociado con un subconjunto 52 de los recursos de hardware de la computadora 10. Por ejemplo, los recursos de un subconjunto 52 sólo son accesibles por el módulo 40, que está alojado en este receptáculo 50.

Por lo tanto, los receptáculos 50 permiten implementar una partición espacial para aislar los módulos de software 40 entre sí. Esta partición se realiza aquí por el sistema de gestión 20, gracias a los mecanismos subyacentes proporcionados por el sistema operativo 18, en particular gracias a la función de separación espacial. Por lo tanto, los módulos 40 pueden ser instalados o desinstalados en caliente, sin afectar el funcionamiento de los otros módulos 40.

La naturaleza precisa de estos mecanismos de partición depende del tipo de sistema operativo 18 utilizado y de la arquitectura de la unidad 12. Aquí, son mecanismos de tipo "partitioning" en lengua inglesa, proporcionados por el sistema operativo 18 en tiempo real. Por ejemplo, alternativamente, el mecanismo de partición se basa en una técnica de dirección virtual conocida como ”Virtual Address Space” en inglés.

Por ejemplo, en el Sistema 20, un módulo 40 no puede acceder a los datos o códigos pertenecientes a otros módulos 40 o a otro software ejecutado en la computadora 10. Un módulo 40 no puede utilizar más recursos de memoria que los que se le asignan estáticamente en el receptáculo 50. Por lo tanto, el receptáculo 50 contribuye al aspecto modular del sistema 20.

Cada receptáculo 50 puede tener varios estados distintos, y concretamente, un estado en el que un módulo 40 está alojado en el receptáculo 50, y otro estado en el que no alberga ningún módulo 40.

En este ejemplo, como se muestra en la figura 4, cada receptáculo 50 puede tener cinco estados diferentes. La referencia "60" se refiere a un estado en el que el receptáculo 50 está vacío. La referencia "62" se refiere a un estado "reservado", en el que no se aloja ningún módulo 40, pero el receptáculo 50 está reservado para la futura adición de un nuevo módulo 40, y por lo tanto no está disponible para alojar ningún módulo 40 que no sea el reservado. La referencia "64" se refiere a un estado "cargado" en el que un módulo 40 está alojado en ese receptáculo 50 pero no se inicia, por ejemplo, porque ese módulo 40 se ha detenido o no ha podido obtener una licencia válida, o su inicialización ha fallado. La referencia "66" se refiere a un estado "inicializado" en el que el módulo 40 está alojado y ha sido iniciado, pero no está en ejecución. Por ejemplo, el módulo 40 está en estado de suspensión, o su ejecución se ha interrumpido debido a un fallo de funcionamiento. Por último, la referencia "68" se refiere a un estado en el que el módulo 40 se está ejecutando.

Las transiciones entre estos estados del receptáculo 50 son proporcionadas aquí por funciones predefinidas. En este ejemplo, la transición entre los estados 66 y 68 se realiza iniciando o, alternativamente, deteniendo la ejecución del módulo 40. La transición entre los estados 66 y 64 se realiza dando la orden de detener el proceso implementado por el módulo 40 o, a la inversa, otorgando una licencia e iniciando este proceso. La transición entre los estados 62 y 64 se logra cargando o, por el contrario, descargando el módulo 40, por ejemplo, de una zona de memoria 14. La transición entre los estados 60 y 62 se logra mediante la validación y aceptación del contrato de servicio del módulo 40 y la asignación de los recursos de hardware del receptáculo 50 para la ejecución de este módulo 40. A la inversa, la transición entre los estados 62 y 60 se logra mediante la liberación de los recursos del receptáculo 50 y la puesta a disposición del receptáculo 50 para otro módulo. La transición de los otros estados al estado 62 también puede hacerse enviando una orden de reinicio al receptáculo 50.

El funcionamiento del sistema 20 se garantiza por un cierto número de componentes de software. Como se muestra en la figura 2, el sistema 20 incluye un componente 22 de supervisión de módulos y un planificador 26. La referencia "24" aquí se refiere a los componentes de software implementados durante el funcionamiento de un módulo 40.

Por ejemplo, el planificador 26 se implementa como una tarea ejecutada por el sistema operativo 18. Por ejemplo, esta ejecución se realiza periódicamente con una frecuencia de ejecución fija.

El sistema 20 también incluye componentes de software para funciones auxiliares.

A modo de ejemplo, el sistema 20 comprende aquí varios componentes auxiliares de software, incluyendo un componente 28 de gestión de licencias, un componente 30 para verificar y gestionar firmas digitales, un componente 32 que proporciona acceso a un servidor de archivos, un componente 34 para gestionar eventos y alarmas, un componente 36 que proporciona acceso a periféricos de hardware de la computadora 10 como sensores de medición integrados en el dispositivo 2, y un componente 38 para gestionar parámetros de configuración. Alternativamente, algunos de estos componentes auxiliares pueden ser omitidos.

En particular, el componente 22 está programado para asegurar la carga de los módulos 40 en los receptáculos 50 y para controlar la ejecución del código 42 correspondiente a través del planificador 26, cuyo funcionamiento se describe a continuación. Por ejemplo, el componente 22 está programado para adquirir los módulos 40, desde la interfaz 16 o desde el sistema de archivos 32, y luego para añadir el código correspondiente a la memoria 14.

En este ejemplo, el componente 22 comprende:

- una interfaz 221 conectada al planificador 26 y capaz de solicitar al planificador 26 que inicie o detenga la ejecución del código 42 de un módulo 40 instalado en un receptáculo 50 ;

- una interfaz 222 para la gestión de los receptáculos 50, en particular capaz de realizar las funciones de carga y descarga de los módulos 40, de registro y eliminación del registro de los módulos 40, y de iniciar y detener la ejecución de un receptáculo 50 ;

- una interfaz 223 en particular capaz de gestionar la adquisición y el acceso al contrato de servicios 44 y de gestionar la adquisición de los módulos 40 del servidor de archivos 32, así como la verificación de su firma digital, cuando proceda.

- una interfaz 224 para acceder al administrador de licencias 28 ;

- una interfaz 225 para acceder al servidor de archivos 32.

El planificador 26 incluye un componente 261 para la gestión de la planificación de tareas y un componente 262 para la supervisión y el control de las tareas en tiempo real que deben ejecutarse.

El componente 24 incluye aquí un componente 241 de gestión de tareas que permite la sincronización de las tareas en tiempo real, un componente 243 de gestión de memoria adaptado para gestionar los accesos del módulo 40 a la memoria 14, una interfaz 244 para el acceso a los componentes auxiliares 34, 36 y 38. El componente 24 también incluye un gestor 242 de tareas, que incluye una interfaz 2421 para acceder al planificador 26 y una interfaz 2422 para gestionar el receptáculo 50.

Las interfaces descritas anteriormente son, por ejemplo, proporcionadas por un conjunto de funciones de interfaz de programación de aplicaciones, también conocidas como "API". Por ejemplo, las conexiones entre las interfaces y/o los componentes de software descritos anteriormente se proporcionan por medios de software para la comunicación entre procesos.

Las figuras 5 y 6 ilustran un aspecto del funcionamiento del planificador 26. Aquí, el planificador 26 es distinto del planificador del sistema operativo 18.

La función del planificador 26 es activar periódicamente la ejecución de las tareas implementadas por los módulos 40 cuando estos módulos están en funcionamiento. Para ello, el planificador 26 utiliza una señal de temporización, como una señal de reloj. Esta señal de temporización es proporcionada aquí por un reloj propio del sistema 20. Alternativamente, esta señal puede ser adquirida desde fuera del sistema 20.

A continuación, a menos que se especifique lo contrario, se entiende que las tareas implementadas por los módulos 40 son tareas en tiempo real (“real-tasks” en inglés).

En este ejemplo, el planificador 26 está programado para procesar las tareas pendientes de ejecución de manera diferente dependiendo de su cronograma de ejecución.

Por ejemplo, cada tarea está asociada a una frecuencia de ejecución predefinida, también llamada cronograma de ejecución. El planificador 26 está programado para autorizar selectivamente la ejecución de las tareas pendientes de ejecución según su frecuencia de ejecución y para asignarles un nivel de prioridad de ejecución que depende de la frecuencia de ejecución definida para esa tarea. El planificador del sistema operativo 18 organiza entonces la ejecución de dichas tareas según este nivel de prioridad de ejecución.

Por ejemplo, se define un primer grupo de tareas, que reúne las tareas a ejecutar con una primera frecuencia de ejecución. El período de ejecución correspondiente a esta frecuencia se denomina aquí período de base y se anota σ_1.

Asimismo, se definen uno o más grupos de tareas, denominados grupos secundarios, agrupando cada uno de ellos las tareas cuya ejecución se pretende lanzar con una frecuencia de ejecución respectiva diferente de la primera frecuencia. Los respectivos valores de periodicidad de ejecución de estos grupos secundarios se señalan aquí de forma genérica mediante la referencia σ_n, donde "n" designa un número entero mayor o igual a 2.

Por ejemplo, las tareas pertenecientes al grupo enésimo secundario, o grupo de orden n, están pensadas para ser ejecutadas con una frecuencia inferior a la frecuencia de ejecución de las tareas del grupo de orden n-1. Por lo tanto, el período σ_n es estrictamente superior al período σ_{n-1}.

En este ejemplo, a modo de ilustración, se define un primer grupo asociado a un primer período σ_1 igual a 10ms, así como cuatro grupos secundarios con sus respectivos períodos de ejecución anotados σ_2, σ_3, σ_4 y σ_5 e iguales respectivamente a 20ms, 100ms, 200ms y 1 segundo.

Como variante, el número de grupos secundarios y sus valores de período asociados pueden ser diferentes de los descritos, y la descripción del planificador 26 y su funcionamiento hecha con referencia a los modos de realización de las figuras 5 y 6 se aplica análogamente a estos otros casos.

Por ejemplo, se genera una señal de temporización específica para cada uno de los grupos para facilitar la sincronización del planificador 26. Cuando los periodos secundarios σ_n son múltiplos del período de base σ_1, pueden generarse señales de temporización secundarias asociadas a los periodos secundarios $\sigma_{n\,a}$ partir de la señal de temporización asociada al período de base σ_1.

Particularmente ventajoso es que los valores de las frecuencias de ejecución, y por lo tanto los valores del período base y/o periodos secundarios descritos anteriormente, se seleccionan de acuerdo a la frecuencia de la corriente eléctrica que fluye a través de los terminales 6 y 8 del dispositivo 2 cuando esta corriente es una corriente alterna. Por ejemplo, el dispositivo 2 se utiliza en una instalación eléctrica en la que la frecuencia de oscilación de la corriente alterna es de 50Hz. Así, en este ejemplo, el período base σ_1 igual a 10ms corresponde a la mitad del período de oscilación de la corriente alterna.

Es ventajoso que la prioridad de ejecución de las tareas pertenecientes a un grupo asociado con una alta frecuencia de ejecución sea mayor que la prioridad de ejecución de las tareas pertenecientes a grupos asociados con una menor frecuencia de ejecución. Por ejemplo, las tareas del grupo de orden n tienen una prioridad de ejecución menor que las del grupo de orden n-1. Las tareas del grupo principal tienen una mayor prioridad de ejecución que las tareas que pertenecen al grupo o grupos secundarios.

En la práctica, esta prioridad es asignada aquí a cada una de las tareas por el planificador 26. Esta prioridad define la prioridad con la que las tareas serán procesadas por el planificador del sistema operativo 18.

En otras palabras, el planificador 26 asigna a dichas tareas un nivel de prioridad de ejecución que depende de la frecuencia de ejecución definida para esta tarea, siendo esta prioridad tanto más alta cuanto mayor sea la frecuencia de ejecución.

Como se muestra en la figura 5, el planificador 26 prioriza las tareas pertenecientes al primer grupo.

Por ejemplo, en el paso 900, el planificador 26 espera la recepción de la señal de temporización asociada al período base σ_1. De hecho, la señal de temporización está fijada preferentemente para que corresponda al período σ_1. En este ejemplo, la señal de temporización se emite cada 10ms.

Una vez que se recibe esta primera señal de temporización, entonces, en un paso 902, el planificador 26 autoriza la ejecución de todas las tareas en tiempo real del primer grupo que están esperando ser ejecutadas.

El procedimiento entonces vuelve al paso 900 y el planificador 26 espera de nuevo a que se reciba la señal de temporización antes de autorizar la ejecución de las nuevas tareas asociadas a este primer grupo que se habrían puesto mientras tanto en espera de ejecución por el resto del sistema.

En la práctica, en este estado, el hecho de que el planificador 26 autorice la ejecución de varias tareas no significa necesariamente que estas tareas serán ejecutadas inmediatamente por la unidad 12, en la medida en que el acceso a la unidad 12 depende del comportamiento del planificador principal gestionado por el sistema operativo 18. No obstante, como el sistema operativo 18 es un sistema en tiempo real, se garantiza que estas tareas se ejecutarán dentro de un marco temporal predefinido. Por lo tanto, la ejecución en tiempo real de estas tareas está garantizada.

A los efectos de esta descripción, debido a los mecanismos de funcionamiento del planificador 26 y del planificador específico del sistema operativo 18 descritos anteriormente, el término "ejecución", cuando se aplica al propio módulo 40, no implica necesariamente que las tareas o procesos definidos por el código ejecutable 42 de este módulo 40 estén siendo ejecutados por la unidad lógica 12. Por ejemplo, por "ejecución del módulo 40" se entiende aquí que el sistema 20 está de acuerdo en hacerse cargo de las tareas correspondientes para su ejecución por la unidad 12, pero que el momento preciso en que las tareas correspondientes son realmente ejecutadas por la unidad 12 depende en última instancia del funcionamiento, en particular, del planificador 26.

La figura 6 ilustra un ejemplo de cómo el planificador 26 procesa las tareas asociadas a uno de los grupos secundarios, como el grupo de orden n.

El procedimiento comienza con un paso inicial 910.

En un paso 912, el planificador 26 comprueba si puede autorizar la ejecución de las tareas del grupo en cuestión. Por ejemplo, el planificador 26 espera la señal de temporización correspondiente al período σ_n.

Si ese es el caso, entonces, en el paso 914, el planificador 26 autoriza la ejecución de las tareas del grupo en cuestión.

En la práctica, en este estado, varias tareas pertenecientes a diferentes grupos han recibido autorización para su ejecución del planificador 26. Es el planificador del sistema operativo 18 quien se encarga de ordenar la ejecución de estas tareas en la unidad 12, según su prioridad, asegurando al mismo tiempo la ejecución en tiempo real de estas tareas.

Una vez lanzada la ejecución de todas las tareas del grupo, el procedimiento vuelve al paso 910 para autorizar la ejecución de las nuevas tareas asociadas a este grupo que habrían sido puestas mientras tanto en espera de ejecución por el resto del sistema.

Así pues, los pasos 910 a 914 de este procedimiento se repiten en un bucle, y se implementan para cada uno de los grupos secundarios, por ejemplo conjuntamente.

Con el diseño descrito anteriormente, el planificador 26 asegura la ejecución de los módulos 40 siguiendo un enfoque secuencial, lo que permite un mayor control sobre la ejecución de estas tareas. Por lo tanto, no es necesario que los programas implementados por el código ejecutable 42 de los módulos de software se basen en un enfoque basado en eventos.

En la práctica, debido a que las tareas del grupo de orden n-1 tienen una mayor prioridad de ejecución que las tareas del grupo de orden n, el planificador del sistema operativo 18 garantiza que las tareas del grupo de orden n no se ejecutarán hasta que las tareas del grupo de orden n-1 hayan terminado de ser ejecutadas.

Sin embargo, como opción, también es posible esperar hasta que las tareas del grupo de orden n-1 hayan completado su ejecución antes de autorizar la ejecución de las tareas del grupo de orden n.

Por lo tanto, el procedimiento anterior se modifica ventajosamente de manera que, en el paso 910, el planificador 26 espera hasta que la unidad 12 haya completado la ejecución de todas las tareas del grupo anterior antes de proceder al paso 912. En otras palabras, la ejecución de las tareas pertenecientes al grupo de orden n sólo se permite cuando se completa la ejecución de las tareas del grupo anterior.

Así, cuando el procedimiento se ejecuta para el grupo enésimo, el planificador 26 espera hasta que todas las tareas del grupo de orden n-1, es decir, las tareas asociadas al periodo σ_{n-1}, hayan terminado de ser ejecutadas. En particular, cuando el procedimiento se ejecuta para el grupo secundario asociado al período σ_2, el planificador 26 espera hasta que todas las tareas del primer grupo hayan terminado de ejecutarse.

La figura 7 muestra un ejemplo de cómo funciona el Sistema 20, para instalar y ejecutar un Módulo 40.

Inicialmente, este módulo 40 no está alojado por ninguno de los receptáculos 50.

Primero, en el paso 1000, el sistema 20 recibe una solicitud de instalación del módulo 40.

Por ejemplo, el usuario envía esta solicitud al sistema 20 a través de la interfaz 16. El módulo 40 es descargado en la memoria por el usuario, si no está ya presente en la memoria. El sistema 20 puede entonces interactuar con este módulo 40, en particular para adquirir el correspondiente contrato de servicios 44.

Luego, en un paso 1002, el sistema 20 comprueba si el contrato de servicio 44 correspondiente es aceptable, en particular teniendo en cuenta los recursos de hardware disponibles que ofrecen los receptáculos 50 que todavía están en estado libre.

Si el correspondiente contrato de servicio 44 se considera aceptable, por ejemplo porque hay al menos un receptáculo 50 libre con recursos 52 suficientes para acomodar las necesidades del módulo 40 como se indica en su contrato de servicio 44, entonces la instalación del módulo 40 se acepta y luego se implementa en un paso 1004 posterior de la instalación del módulo 40 en el receptáculo 50 libre.

De lo contrario, la instalación del módulo 40 es rechazada y el procedimiento termina en el paso 1006.

Una vez que el módulo 40 está correctamente instalado en el receptáculo 50, su ejecución se inicia en el paso 1008.

En un paso de 1010 se ejecutan las instrucciones 42 del módulo 40, por ejemplo de forma cíclica mediante la intervención del planificador.

A lo largo de este paso 1010, durante la ejecución del módulo 40, el sistema 20 asegura, aquí en un paso 1012, que el módulo 40 respeta su contrato de servicio 44. En otras palabras, el sistema 20 verifica que el módulo 40 no reclama ni consume recursos de hardware de la computadora 10 que sean mayores que las necesidades establecidas en su contrato de servicio 44.

Si el módulo cumple con el contrato de servicio, entonces, en un paso 1014, se permite que la ejecución del módulo 40 continúe normalmente. El paso 1012 se ejecuta de nuevo, por ejemplo, periódicamente, siempre que el paso 1010 continúe.

Por el contrario, si el paso 1012 revela que el contrato de servicio no es respetado por el módulo 40, entonces el sistema 20 implementa una estrategia de recuperación en un paso 1016.

El paso 1016 permite reiniciar el sistema 20 en un estado estable lo más rápidamente posible, de modo que el funcionamiento de los otros módulos 40 no se vea afectado negativamente por el comportamiento defectuoso del módulo 40 que no ha respetado su contrato de servicio.

Como se explica en la referencia a la figura 14, esta estrategia de recuperación puede ir acompañada de un cierre forzado del módulo 40 defectuoso.

Finalmente, la ejecución del módulo 40 termina normalmente, por ejemplo porque el módulo 40 ha completado normalmente su misión o porque ha recibido una orden de parada. El paso 1010 entonces termina y el procedimiento termina en el paso 1018.

Los pasos 1000 a 1018 descritos anteriormente se implementan aquí en paralelo para cada uno de los módulos 40 que están actualmente en ejecución.

La figura 8 muestra un ejemplo de cómo implementar los pasos 1002, 1004 y 1008 para verificar el contrato de servicio 44 de un módulo 40 cuando se ha solicitado su instalación.

Inicialmente, en un paso 1020, el contenido del contrato de servicio 44 del módulo 40 es cargado por el componente 30 y luego es leído automáticamente por el administrador 223 del componente 22.

En un paso 1022, el sistema 20 adquiere las declaraciones registradas en el contrato de servicio 44, en particular las relativas a los recursos de computación requeridos por el módulo 40.

Estas declaraciones se comparan, en un paso de 1024, con los recursos disponibles ofrecidos por los receptáculos 50 del sistema 20 que se declaran como disponibles, es decir, que están en estado vacío.

Si no se encuentra ningún receptáculo adecuado, por ejemplo porque ninguno de los módulos 50 libres puede ofrecer los recursos solicitados por el contrato de servicio 44, entonces el procedimiento de verificación termina, en el paso 1026, con el rechazo del contrato de servicio. Por ejemplo, el paso 1006 de negarse a instalar el módulo 40 es entonces implementado.

Por el contrario, si se dispone de un receptáculo 50 adecuado, entonces en el paso 1028, ese receptáculo 50 está reservado. Por ejemplo, el gestor 223 actualiza automáticamente una lista de los receptáculos 50 disponibles en el sistema 20.

Luego, en el paso 1030, el módulo 40 se carga en el receptáculo 50 correspondiente. Ventajosamente, la compatibilidad del módulo 40 puede ser comprobada en esta ocasión.

Si el módulo 40 se identifica como no compatible, entonces el procedimiento en el paso 1032 rechazando el módulo. Por ejemplo, se implementa el paso 1006 de rechazar la instalación del módulo 40.

Opcionalmente, si el módulo 40 es aceptado, entonces su licencia se comprueba automáticamente en el paso 1036, aquí con la ayuda de la interfaz 224 y el componente 28. Esto permite verificar el derecho a usar el módulo 40.

Si en el paso 1038 la licencia del módulo 40 es identificada como inválida o ausente, entonces en el paso 1040 el procedimiento termina, con el rechazo del Módulo 40. Por ejemplo, el paso 1006 de rechazar la instalación del Módulo 40 es implementado nuevamente.

Por el contrario, si la licencia es aceptada, entonces el procedimiento continúa.

Luego, durante un paso 1042, el módulo 40 se inicializa automáticamente, es decir, se pone en un estado que permite iniciar más tarde la ejecución del programa asociado al código 42. Por ejemplo, el gestor 222 ordena la inicialización del receptáculo 50.

Si la inicialización falla por cualquier razón, entonces el procedimiento termina en un paso 1046 rechazando el módulo 40. Por ejemplo, el paso 1006 de rechazar el módulo 40 se implementa de nuevo.

Si la inicialización procede normalmente, entonces el receptáculo 50 está en el estado 66 después del paso 1044 y las tareas recurrentes de código 42 están listas para ser ejecutadas.

En este ejemplo, el procedimiento incluye los pasos 1048 y 1050 para actualizar la información de seguimiento del estado del sistema 20, por ejemplo, la proporcionada por el componente 22, para reflejar este cambio de estado.

El procedimiento normalmente termina en un último paso 1052.

La figura 9 es un ejemplo de la implementación del paso 1012 de verificación del cumplimiento del contrato de servicio 44 de un módulo 40 durante la ejecución del módulo 40.

Este ejemplo se refiere más precisamente a la implementación de la verificación del cumplimiento del tiempo de ejecución de una tarea perteneciente a uno de los grupos de tareas, perteneciendo aquí, como ejemplo ilustrativo, al grupo secundario asociado a la duración σ_2 igual a 20ms.

En un paso de 1060, el sistema 20 espera el final del ciclo de ejecución asociado a las tareas de grupo. El final de este ciclo de ejecución viene dado, por ejemplo, por la recepción de una señal de temporización correspondiente.

Una vez que el sistema 20 ha detectado el final de este ciclo de ejecución, entonces se implementa un paso 1062 para cada tarea del grupo cuya ejecución debería haber terminado.

Este paso 1062 tiene un sub-paso 1064, en el que el sistema 20 determina automáticamente el tiempo de ejecución real de esta tarea durante el ciclo de ejecución. Esta determinación se basa en la información que procede del núcleo del sistema operativo 18. A continuación se describe un ejemplo de esa determinación con referencia a las figuras 11 a 13.

Una vez adquirido el tiempo de ejecución real, entonces, en un sub-paso 1066, este tiempo de ejecución real se compara automáticamente con el tiempo de ejecución máximo declarado en el contrato de servicio 44.

Si el tiempo de ejecución real es mayor que el tiempo máximo de ejecución declarado en el contrato de servicio 44, entonces se considera que el módulo 40 ha incumplido su contrato de servicio 44. Por lo tanto, en un paso 1068, se implementa una estrategia de recuperación para evitar que el Módulo 40 vuelva a incumplir su contrato de servicio 40 para no penalizar el funcionamiento de los otros módulos 40. Por ejemplo, se implementa el paso 1016 descrito anteriormente.

Ventajosamente, en un paso 1070, el incumplimiento se registra en un archivo de registro.

Luego, en el paso 1072, se completa el paso 1062 y luego se repite para procesar la siguiente tarea dentro del grupo de tareas.

Cuando no quedan más tareas ejecutadas en el grupo correspondiente, termina el paso 1062. El procedimiento se reinicia en el paso 1060 para esperar el final del siguiente ciclo.

Por el contrario, si el paso 1066 muestra que el módulo 40 ha cumplido su contrato de servicio 44 en términos de recursos de computación durante este ciclo de computación, entonces el paso 1072 se implementa directamente. Cuando no quedan más tareas ejecutadas en el grupo correspondiente, entonces termina el paso 1062.

Como alternativa, este paso 1012 puede realizarse de manera diferente, en particular para verificar el cumplimiento del contrato de servicios para otros recursos de hardware distintos del tiempo de ejecución. Por ejemplo, el paso 1012 se refiere genéricamente a una serie de procedimientos de verificación realizados conjuntamente para verificar el cumplimiento de diferentes aspectos del contrato de servicios de un mismo módulo.

Los pasos 1080 a 1090 descritos anteriormente se implementan aquí para cada uno de los módulos 40 independientemente de cada uno de ellos. En particular, se entiende que en el sistema 20 pueden implementarse varios procedimientos similares al de la figura 9 para comprobar el cumplimiento del contrato de servicios por tareas pertenecientes a distintos grupos

En las figuras 10 a 13 se muestra un ejemplo de un procedimiento implementado para medir el tiempo de ejecución de una tarea ejecutada por un módulo 40 cargado en el sistema 20, por ejemplo en el transcurso del paso 1064 descrito anteriormente.

Los pasos implementados en los procedimientos de las figuras 10 a 13 se describen aquí secuencialmente para facilitar su comprensión. Sin embargo, en la práctica, los procedimientos correspondientes a los diagramas de flujo de las figuras 10 a 13 se implementan en paralelo entre sí.

Como se muestra en la figura 10, el procedimiento consiste en un primer subconjunto de pasos. En un paso 1080, el sistema monitoriza el planificador del sistema operativo 18 para detectar un posible cambio de contexto (“context switch” en ingés) y espera hasta que dicho cambio de contexto se produzca.

Cuando se produce tal cambio de contexto, entonces, en un paso 1082, el sistema comprueba si una de las tareas implicadas en este cambio de contexto, ya sea como tarea entrante o como tarea saliente, es una tarea en tiempo real que pertenece a uno de los grupos de tareas descritos anteriormente. Si no es así, entonces el procedimiento vuelve al paso 1080.

Si este es el caso, por el contrario, entonces en un paso del 1084, el sistema 20 determina automáticamente si esta tarea en tiempo real es entrante, es decir, accede a la unidad lógica 12, o si es saliente, es decir, deja de ser ejecutada por la unidad lógica 12.

Si se determina que la tarea en tiempo real es de entrada, entonces, en un paso de 1086, el sistema 20 registra automáticamente el instante en que la tarea fue de entrada, es decir, el momento en que la tarea tuvo acceso a la

unidad lógica 12. Por ejemplo, este instante se almacena en una estructura de datos dedicada dentro de la memoria 14. El procedimiento termina para esta tarea en el paso 1088 y luego regresa al paso 1080 para ser implementado nuevamente.

Si, por el contrario, se determina que la tarea en tiempo real es saliente, entonces en un paso de 1090, el sistema 20 calcula automáticamente el tiempo que ha transcurrido desde la última entrada. Esta vez se denota aquí como T1. En este paso 1090, el sistema 20 determina si la tarea se adelantó como resultado de una rutina de interrupción ("Interrupt Service Routine" en inglés). Si este es el caso, el sistema recupera el tiempo de T_{isr} durante el cual esta tarea fue adelantada. El cálculo de esta duración del T_{isr} se describe a continuación con referencia a la figura 13. Una vez que se conoce la duración del T_{isr}, el sistema 20 calcula la diferencia "T_1 - T_{isr}" y luego añade esta diferencia al valor del tiempo de ejecución de esta tarea. El valor del tiempo de ejecución se almacena, por ejemplo, en un acumulador asociado a la unidad lógica 12, al que se puede acceder mediante el sistema 20.

Paralelamente, como se muestra en la figura 11, el sistema 20 espera en un paso 1110 la recepción de una señal de sincronización asociada al tiempo de ejecución umbral correspondiente al grupo de tareas. Entonces, una vez que se recibe esta señal, el sistema 20 reinicia automáticamente el acumulador para todas las tareas del grupo. Los pasos 1110 y 1112 se repiten en un bucle.

Esta figura 11 describe el procedimiento para el grupo de tareas enésimo. En la práctica, los pasos 1110 y 1112 se implementan por separado para cada una de las n familias de tareas definidas para el Sistema 20.

Como se muestra en la figura 12, en paralelo, en un paso 1120, el sistema 20 espera la recepción de la información de que se ha producido una condición de excepción asociada a la unidad 12. Esta información es, por ejemplo, una señal proporcionada por una rutina de interrupción.

Entonces, en un paso 1122, el sistema 20 almacena, por ejemplo en la memoria 14, la información de la marca de tiempo sobre la fecha de recepción de esta condición de excepción y sobre el hecho de que tal condición de excepción ha ocurrido.

En respuesta, en el paso 1124, el planificador 26 instruye al planificador del sistema operativo 18 para ejecutar una tarea específica, anotada T0. Esta tarea T0 corresponde a una tarea vacía con la mayor prioridad de ejecución. Por ejemplo, esta prioridad se define estáticamente como la más alta en todas las circunstancias.

El paso 1120 se ejecuta de nuevo para esperar una futura condición de excepción.

Al mismo tiempo, como se muestra en la figura 13, en un paso 1130, el sistema 20 monitoriza las tareas gestionadas por el planificador del sistema operativo 18, en particular para detectar las tareas que son ejecutadas por la unidad 12.

Tan pronto como el sistema 20 detecta que el planificador del sistema operativo 18 desencadena la ejecución de una tarea, entonces, en un paso 1132, el sistema 20 verifica automáticamente si la tarea entrante es la tarea específica T0. Si este es el caso, entonces en el paso 1134, el sistema 20 calcula y registra el tiempo T_{isr}, que ha transcurrido desde la última condición de excepción detectada. Este cálculo se realiza utilizando, por ejemplo, la información de la marca de tiempo registrada en el paso 1122. El procedimiento entonces vuelve al paso 1100.

Si, por el contrario, la tarea entrante no se identifica como la tarea específica T0, entonces el procedimiento vuelve al paso 1130.

Por lo tanto, el valor del tiempo de ejecución contenido en el acumulador corresponde al tiempo de ejecución real de la tarea. Este valor es, por ejemplo, recogido por el sistema 20, antes del paso de reinicio 1112, a fin de implementar el mencionado paso 1066.

El procedimiento de las figuras 10 a 13 se implementa ventajosamente cuando el sistema 20 no puede acceder directamente al núcleo del sistema operativo 18. Permite así disponer de una información fiable sobre el tiempo que tarda la unidad 12 en ejecutar las tareas, basándose en la información recogida en el espacio de usuario del sistema operativo 18.

La figura 14 es un ejemplo de implementación del paso 1016 para implementar una estrategia de recuperación cuando un módulo 40 no cumplió su contrato de servicio durante su ejecución. Este ejemplo del paso 1016 de recuperación se describe para el caso en que el incumplimiento del contrato de servicios está relacionado con el tiempo de ejecución.

Primero, en un paso 1150, todas las tareas en ejecución del módulo 40 se interrumpen y se eliminan, por ejemplo enviando una señal de parada de tipo "kill".

Luego, en el paso 1152, el número de intentos de reinicio del módulo 40 se compara con un umbral máximo predefinido.

Si el número de intentos es menor que el umbral máximo, entonces en el paso 1154 se intenta reiniciar el módulo 40. Por ejemplo, se crean y se restablecen nuevas tareas correspondientes a las tareas eliminadas en el paso 1150. El contador que indica el número de intentos de reinicio se incrementa en uno.

Entonces, en un paso 1156, las tareas así reiniciadas son asumidas por el planificador para asegurar su ejecución.

El procedimiento termina en el paso 1158.

En cambio, si en el paso 1152 el número de intentos es mayor que el umbral máximo, entonces en el paso 1160 se detiene la ejecución del módulo 40.

Este paso de recuperación 1016 garantiza que en caso de incumplimiento del contrato de servicio de un módulo 40, causando un consumo excesivo de recursos de hardware por parte de este módulo 40, los otros módulos 40 no son penalizados por más de un ciclo en su tiempo de ejecución. En otras palabras, el paso 1016 de recuperación garantiza la cohabitación de varios módulos 40 mientras se preserva el comportamiento en tiempo real.

Alternativamente, el paso 1016 puede realizarse de manera diferente.

Por ejemplo, el paso 1016 puede implicar, tras la detección de un incumplimiento del contrato de servicio por parte de un módulo 40, el disparo de una alarma para advertir al usuario de un mal funcionamiento y luego se detiene el módulo 40 defectuoso.

Según otra alternativa, la detención del módulo 40 puede ser seguida por el uso de la siguiente franja horaria para ejecutar los otros módulos 40 que no pudieron ser ejecutados correctamente debido al incumplimiento. Una vez completado este ciclo, se puede intentar reiniciar el módulo 40 defectuoso, por ejemplo, como se describe en la figura 14.

Por otra parte, el paso 1016 también puede realizarse de manera diferente cuando el incumplimiento del contrato de servicios se refiere a un recurso de hardware distinto del tiempo de ejecución.

Según otras variantes, la estrategia es diferente dependiendo del valor de criticidad del módulo defectuoso declarado en el contrato de servicio. Esto se debe a que las funciones relacionadas con la protección del dispositivo 2 tienen una mayor criticidad que otras funciones, como las de medición. Según la estrategia general de funcionamiento del dispositivo 2, puede ser preferible, en caso de incumplimiento, aplazar ciérrela detención de los módulos que implementan estas funciones el mayor tiempo posible, por ejemplo, para garantizar la continuidad del servicio del dispositivo 2. En este caso, si se detecta más de un incumplimiento, se detendrá primero el módulo 40 con menor criticidad, por ejemplo, utilizando los pasos anteriores.

En otros casos, se puede dar prioridad a la seguridad forzando el dispositivo 2 a una posición de seguridad, por ejemplo forzando la apertura los contactos separables del bloque de conmutación 4.

Gracias a la invención, es posible conciliar tanto el aspecto modular del software incorporado en la computadora 10 como el respeto de las restricciones de ejecución en tiempo real de este software.

Como los módulos de software 40 tienen una estructura modular y pueden ser instalados o desinstalados dinámicamente, la gestión del software de la computadora 10 del dispositivo 2 se facilita a los usuarios. Esta estructura modular también ofrece más posibilidades para seguir desarrollando el funcionamiento del dispositivo 2 a lo largo del tiempo. Además, al preservar el funcionamiento en tiempo real del software incorporado, la seguridad operativa del dispositivo 2 no se ve afectada.

Como se ha explicado anteriormente, la estructura modular de los módulos 40 está parcialmente permitida por los receptáculos 50. Esta estructura modular también está permitida por el contrato de servicios 44, así como por los mecanismos de verificación del cumplimiento del contrato de servicios, ya sea en la instalación o en el curso de la ejecución, o ambos, y por la implementación de la estrategia de recuperación en caso de incumplimiento del contrato de servicios. Los contratos de servicio 44 y el funcionamiento del planificador 26 también permiten asegurar el funcionamiento en tiempo real de los módulos 40.

Los modos de realización y las variantes consideradas anteriormente pueden combinarse entre sí para generar nuevos modos de realización.

REIVINDICACIONES

1. Procedimiento de gestión de módulos de software integrados (40) para una computadora electrónica integrada (10) de un dispositivo eléctrico de conmutación (2) de una corriente eléctrica, en el que este procedimiento comprende los siguientes pasos:

a) adquisición (1020), por un sistema de gestión (20) de módulos de software de la computadora electrónica (10) del dispositivo de conmutación (2), de un módulo de software (40) que comprende un código ejecutable (42) y un contrato de servicio (44) en el que se declaran los recursos de hardware que requiere el código ejecutable (42) durante su ejecución por la computadora electrónica;

b) instalación (1004) del módulo de software (40) dentro de un receptáculo de acogida (50) por el sistema de gestión (20), perteneciendo dicho receptáculo de acogida a un conjunto de receptáculos de acogida (50) destinados cada uno a formar un entorno para ejecutar un módulo de software (40), teniendo cada receptáculo (50) para este propósito una ubicación de memoria definida estáticamente dentro de una memoria (14) de la computadora electrónica (10) y estando asociada con un subconjunto (52) de recursos de hardware de la computadora electrónica (10);

c) Ejecución (1010) del módulo de software (40) por la computadora electrónica (10);

en el que el paso c) comprende, durante la ejecución del módulo de software (40), un paso de comprobación (1012), mediante el sistema de gestión (20), de si la ejecución del módulo de software (40) cumple con el contrato de servicio (44) de este módulo de software (40), en el que se permite que dicha ejecución continúe si se cumple el contrato de servicio, y en el que, si se identifica la ejecución del módulo de software (40) como no conforme al contrato de servicio, se establece un paso de recuperación (1016) con el fin de interrumpir la ejecución del módulo de software (40) por parte de la computadora electrónica (10),
y en el que, en el paso c), la ejecución del código (42) comprende la ejecución de una pluralidad de tareas en tiempo real por parte de una unidad lógica de cálculo (12) de la computadora electrónica (10), estando asociada cada tarea a una frecuencia de ejecución predefinida, y en el que la ejecución de las tareas por la unidad lógica de cálculo (12) está regulada por un primer planificador (26) implementado por el sistema de gestión (20) y luego por un segundo planificador implementado por un sistema operativo en tiempo real (18) de la computadora electrónica (10), autorizando el primer planificador (26) selectivamente la ejecución de las tareas pendientes de ejecución en función de su frecuencia de ejecución y les asigna un nivel de prioridad de ejecución para el segundo planificador que depende de la frecuencia de ejecución definida para esta tarea.

2. Procedimiento según la reivindicación 1, **caracterizado porque** el procedimiento comprende, antes del paso b), un paso previo de verificación (1002) del contrato de servicios (44) del módulo de software (40), rechazándose la instalación del módulo de software (40) si los recursos de hardware requeridos por el contrato de servicios (44) no son compatibles con los recursos de hardware ofrecidos por los receptáculos (50) disponibles.

3. Procedimiento según la reivindicación 1 o 2, **caracterizado porque**, paralelamente al paso c), se está ejecutando también al menos un segundo módulo de software (40) diferente instalado en otro receptáculo diferente (50), y se está ejecutando también un segundo paso de comprobación (1012) de si la ejecución de este segundo módulo de software (40) cumple con el contrato de servicios (44) de este segundo módulo de software (40).

4. Procedimiento según cualquiera de las afirmaciones anteriores, **caracterizado porque** un primer grupo de tareas comprende tareas asociadas a una primera frecuencia de ejecución predefinida, **porque** un segundo grupo de tareas comprende tareas asociadas a una segunda frecuencia de ejecución predefinida que es inferior a la primera frecuencia de ejecución,
porque, durante la ejecución del módulo de software (40), es activada (902) la ejecución de las tareas pertenecientes al primer grupo por el primer planificador (26) cada vez que se recibe (900) una señal de temporización periódica,
y **porque** la ejecución de las tareas pertenecientes al segundo grupo es disparada (914) periódicamente por el primer planificador (26), teniendo las tareas pertenecientes al segundo grupo una prioridad menor que las tareas pertenecientes al primer grupo.

5. Procedimiento según cualquiera de las reivindicaciones anteriores, **caracterizado porque** el dispositivo de conmutación (2) está conectado a una instalación eléctrica para interrumpir una corriente eléctrica alterna dentro de esta instalación eléctrica, seleccionándose las frecuencias de ejecución en función de la frecuencia de la corriente eléctrica alterna.

6. Procedimiento según cualquiera de las reivindicaciones anteriores, **caracterizado porque** la estrategia de recuperación implementada en el paso de recuperación (1016) en caso de incumplimiento del contrato de servicios (44) de un módulo de software (40) incluye la detención de la ejecución del módulo de software (40).

7. Procedimiento según cualquiera de las reivindicaciones anteriores, **caracterizado porque** la estrategia de recuperación implementada en el paso de recuperación (1016) en caso de incumplimiento del contrato de servicio (44) de un módulo de software (40) se selecciona en función de un valor de criticidad del código ejecutable (42) declarado en el contrato de servicio (44) de este módulo de software (40).

8. Dispositivo de conmutación (2) de una corriente eléctrica que comprende una computadora electrónica integrada (10) adecuada para controlar el funcionamiento del dispositivo de conmutación (2) y diseñado para recibir módulos de software integrados (40),
comprendiendo la computadora electrónica (10) un conjunto de receptáculos de acogida (50) cada uno diseñado para formar un entorno para ejecutar un módulo de software (40), comprendiendo cada receptáculo (50) para este propósito una ubicación de memoria definida estáticamente dentro de una memoria (14) de la computadora electrónica (10) y que está asociada a un subconjunto (52) de recursos de hardware de la computadora electrónica (10),
implementando además esta computadora electrónica (10) un sistema de gestión (20) programado para implementar las pasos:

a) la adquisición de un módulo de software (40) que comprende el código ejecutable (42) y un contrato de servicios (44) en el que se declaran los recursos de hardware que requiere el código ejecutable (42) cuando es ejecutado por la computadora electrónica ;

b) la instalación (1004) del módulo de software (40) en uno de los receptáculos (50);

c) la ejecución (1010) del módulo de software (40) por la computadora electrónica (10), comprendiendo la paso c), durante la ejecución del módulo de software (40), una paso consistente en verificar (1012) si la ejecución del módulo de software (40) cumple con el contrato de servicio (44) para este módulo de software (40), se permite que la ejecución continúe si se cumple el contrato de servicio y, si se identifica la ejecución del módulo de software (40) como no conforme al contrato de servicio, se establece una paso de recuperación (1016) con el objetivo de interrumpir la ejecución del módulo de software (40) por la computadora electrónica (10)

y en el paso c), la ejecución del código (42) comprende la ejecución de una pluralidad de tareas en tiempo real por una unidad de computación lógica (12) de la computadora electrónica (10), estando cada tarea asociada a una frecuencia de ejecución predefinida, y en el que la ejecución de las tareas por la unidad de computación lógica (12) está regulada por un primer planificador (26) implementado por el sistema de gestión (20) y luego por un segundo planificador implementado por un sistema operativo (18) en tiempo real de la computadora electrónica (10), autorizando el primer planificador (26) selectivamente la ejecución de las tareas pendientes de ejecución en función de su frecuencia de ejecución y les asigna un nivel de prioridad de ejecución para el segundo planificador que depende de la frecuencia de ejecución definida para esta tarea.

9. Dispositivo de conmutación (2) según la reivindicación 8, **caracterizado porque** la computadora electrónica (10) está programada además para aplicar, antes del paso b), un paso de verificación preliminar (1002) del contrato de servicios (44) del módulo de software (40), rechazándose la instalación del módulo de software (40) si los recursos de hardware requeridos por el contrato de servicios (44) no son compatibles con los recursos de hardware ofrecidos por los receptáculos (50) disponibles.

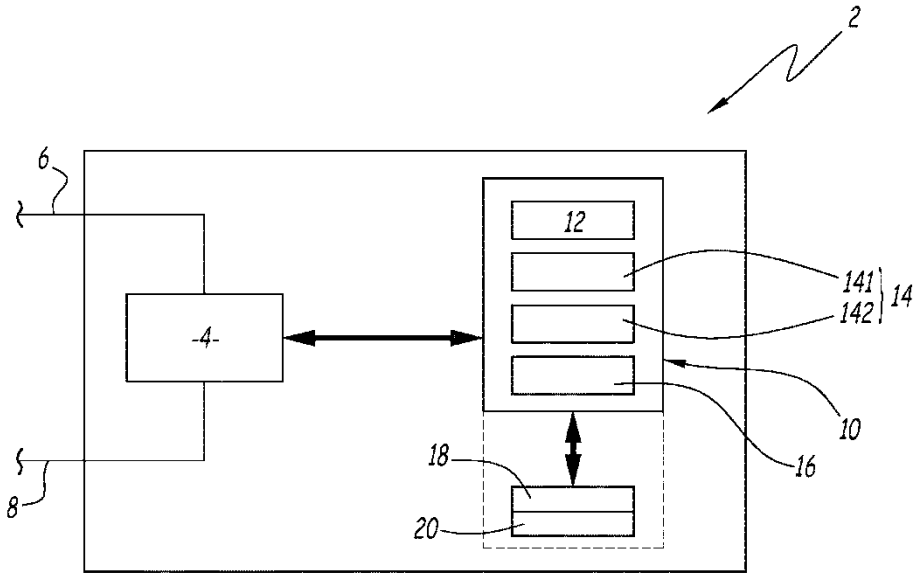

Fig.1

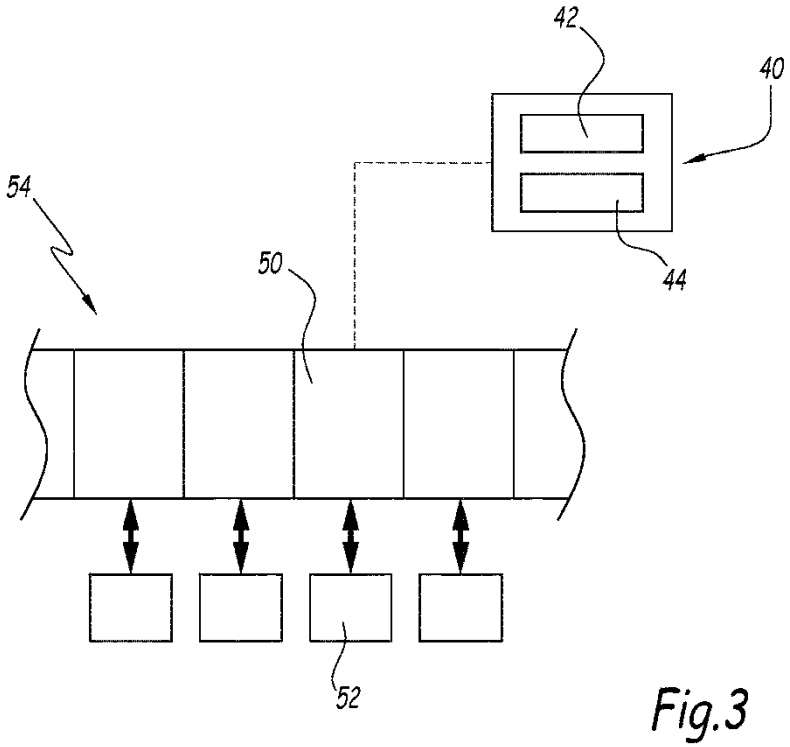

Fig.3

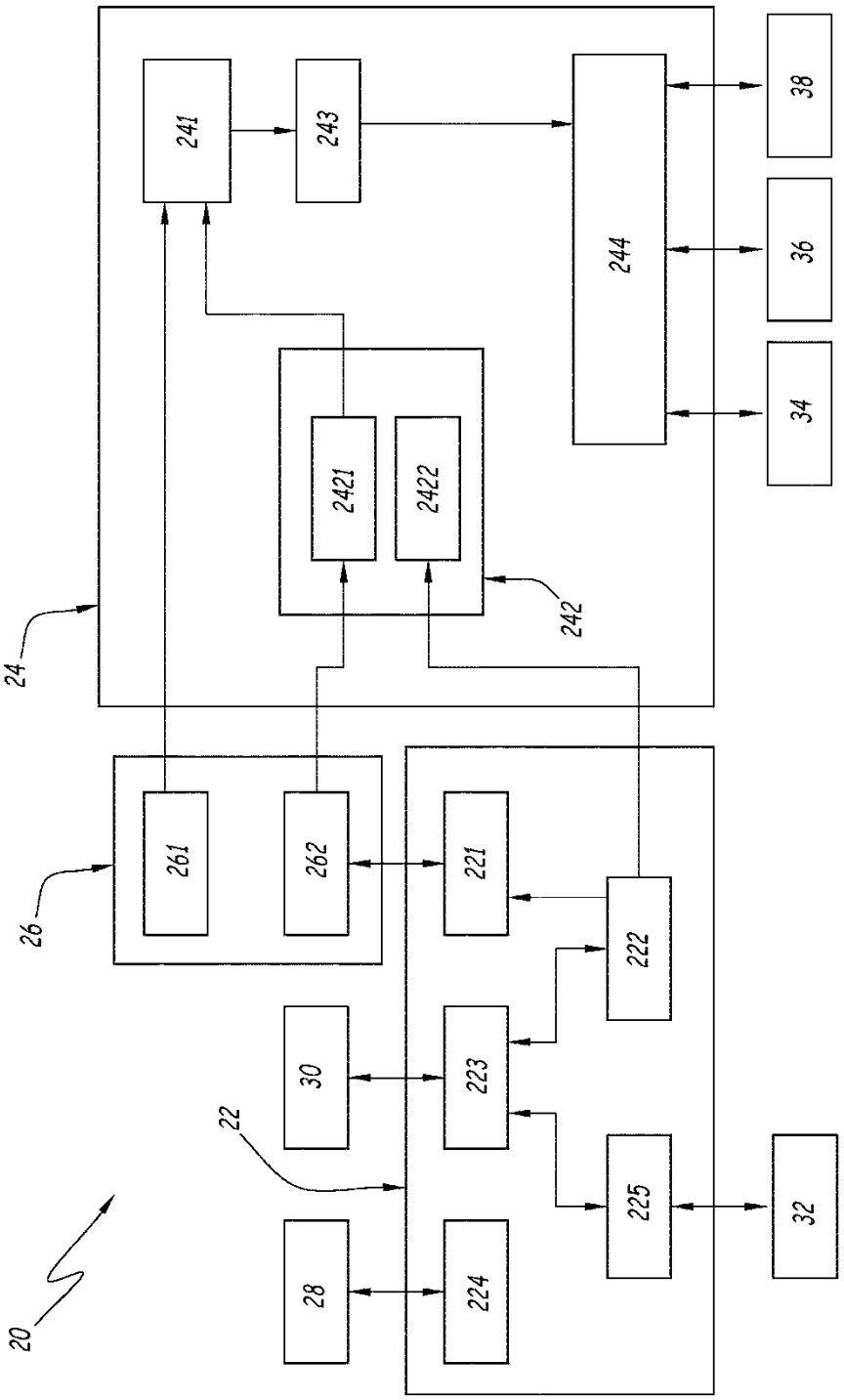
241
243
244
38
36
34
2421
2422
242
24
261
262
221
222
26
30
223
22
225
32
28
224
20

Fig.2

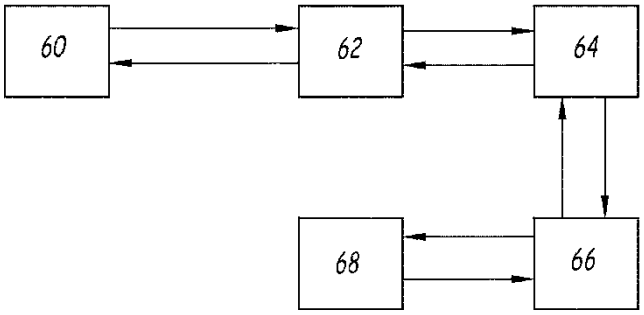

Fig.4

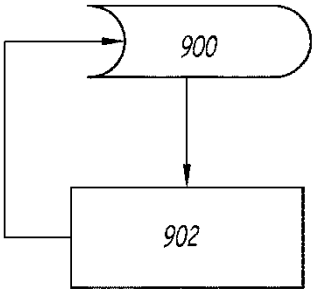

Fig.5

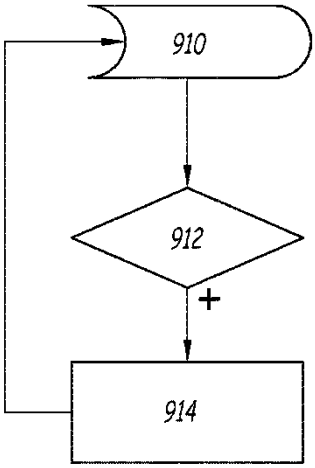

Fig.6

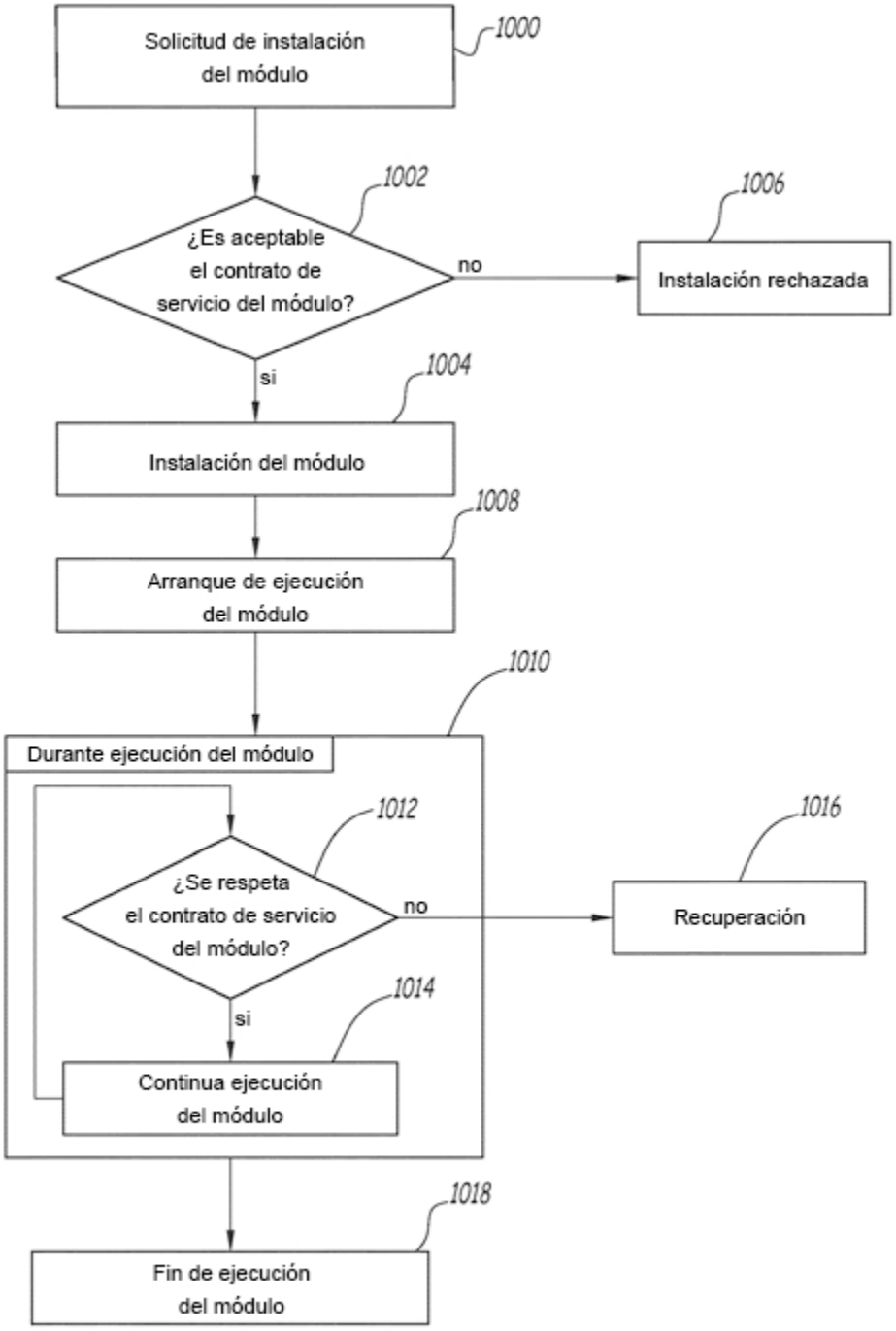
Solicitud de instalación del módulo
1000
1002
¿Es aceptable el contrato de servicio del módulo?
no
1006
Instalación rechazada
si
1004
Instalación del módulo
1008
Arranque de ejecución del módulo
1010
Durante ejecución del módulo
1012
¿Se respeta el contrato de servicio del módulo?
no
1016
Recuperación
si
1014
Continua ejecución del módulo
1018
Fin de ejecución del módulo

Fig.7

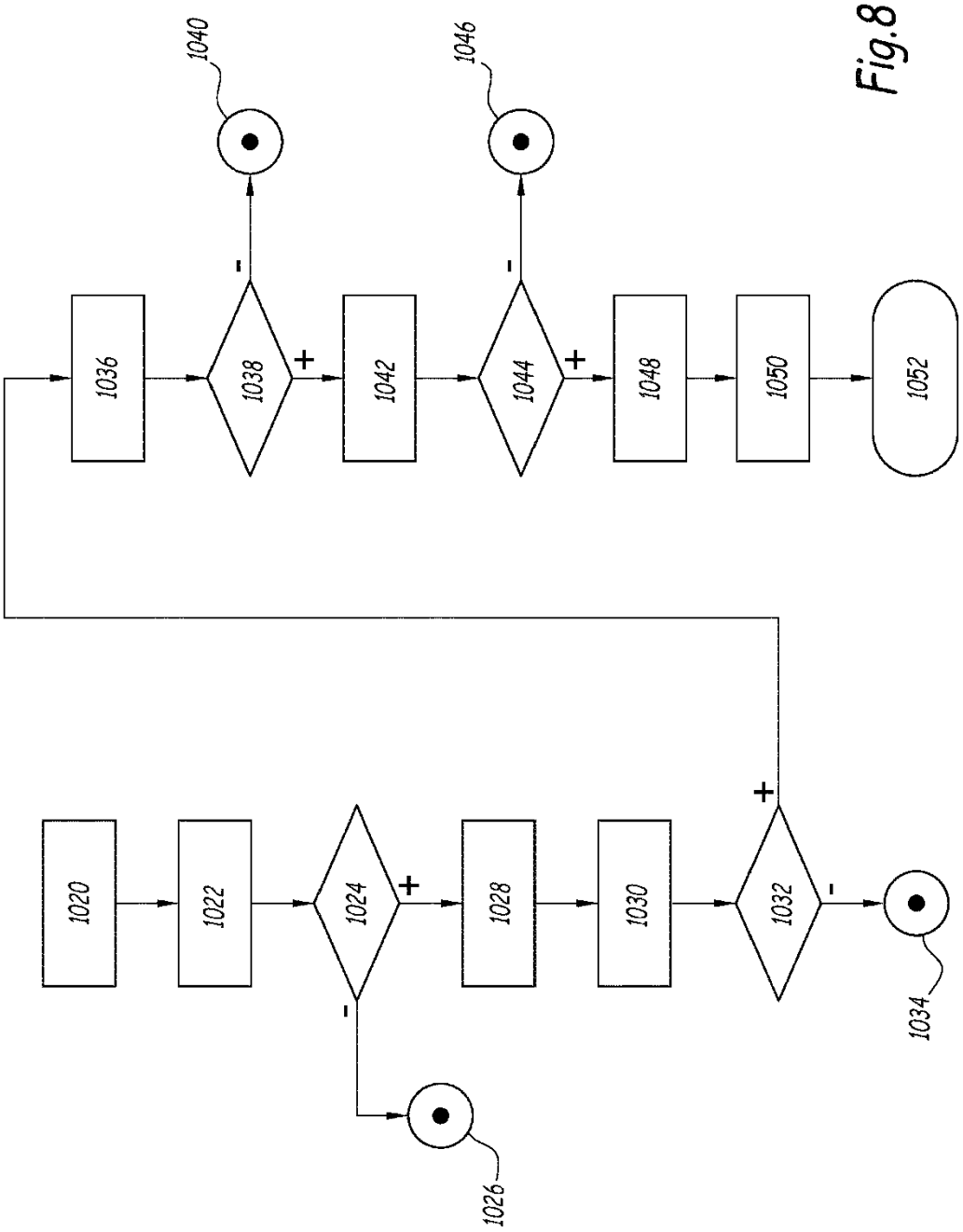
1040
1046
1036
1038
1042
1044
1048
1050
1052
1020
1022
1024
1028
1030
1032
1034
1026
+
-

Fig.8

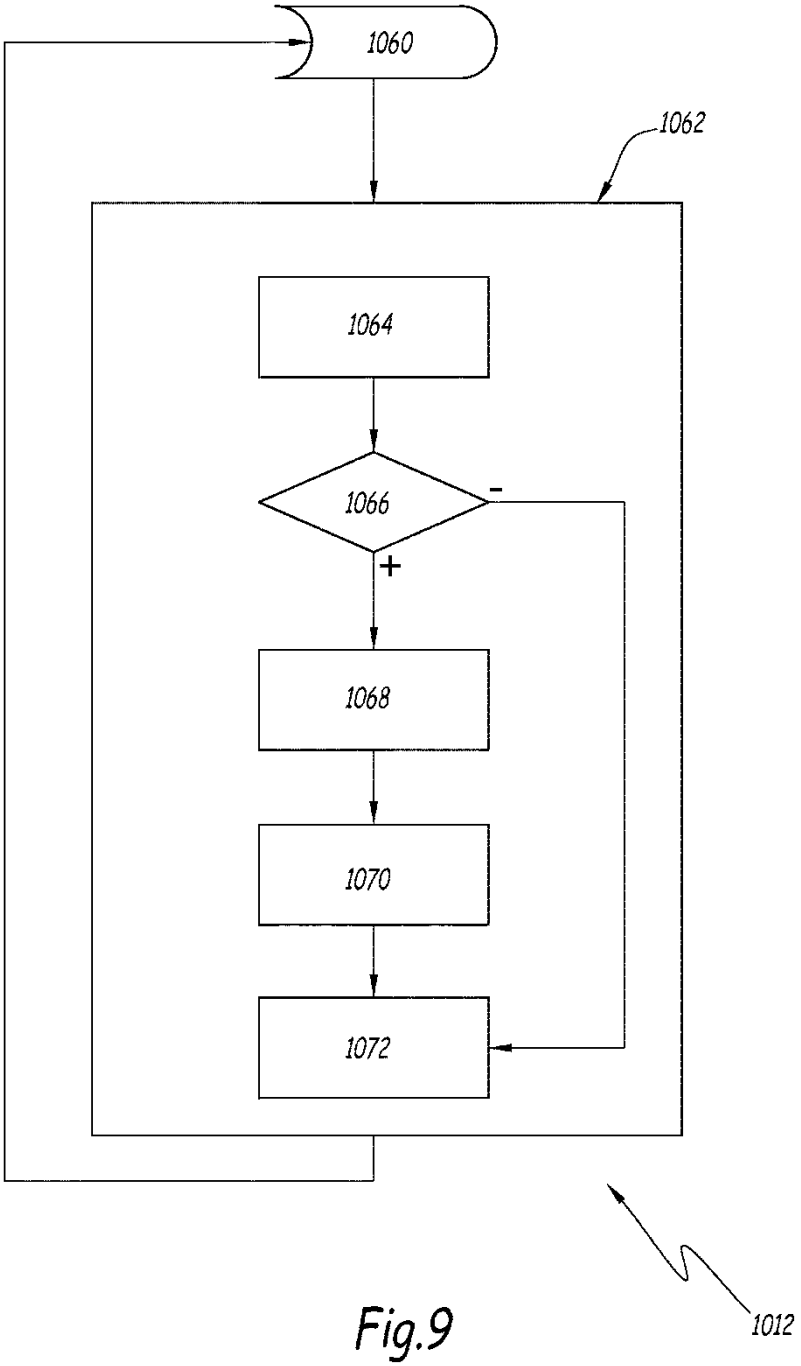

Fig.9

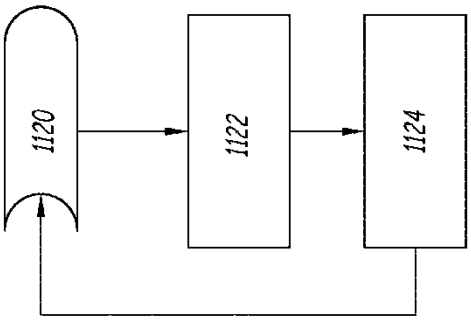

Fig.12

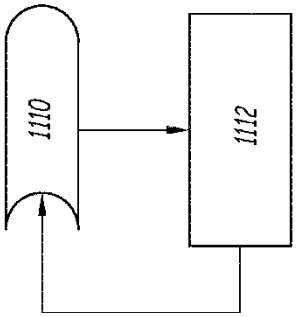

Fig.11

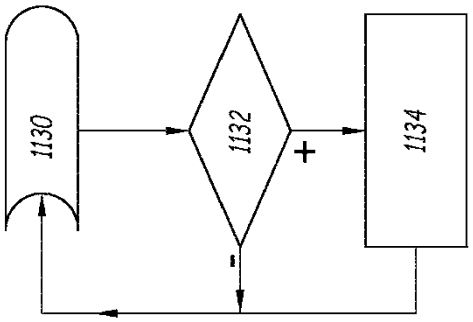

Fig.13

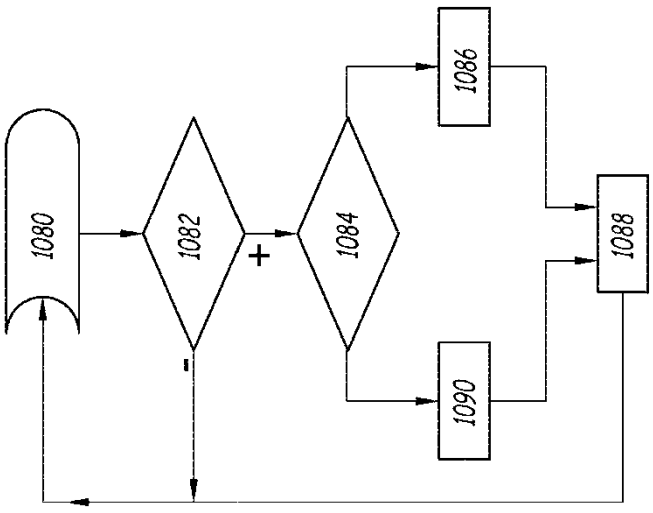

Fig.10

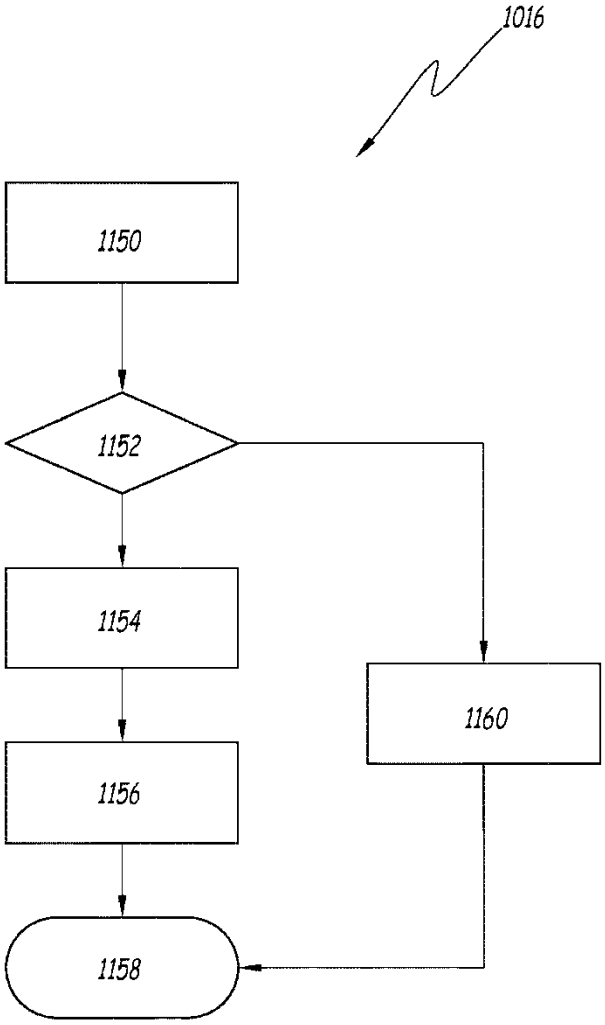

Fig.14