(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 702 470**

(51) Int. CI.:

G06K 19/073 (2006.01)
G07F 7/10 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **03.11.2010 PCT/EP2010/006693**

(87) Fecha y número de publicación internacional: **12.05.2011 WO11054498**

(96) Fecha de presentación y número de la solicitud europea: **03.11.2010 E 10781422 (0)**

(97) Fecha y número de publicación de la concesión europea: **18.07.2018 EP 2497055**

(54) Título: **Objeto portátil seguro**

(30) Prioridad:

05.11.2009 FR 0905312

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
01.03.2019

(73) Titular/es:

GEMALTO SA (100.0%)
6 rue de la Verrerie
92190 Meudon, FR

(72) Inventor/es:

REGNAULT, NICOLAS y
VETILLARD, ERIC

(74) Agente/Representante:

CASANOVAS CASSÁ, Buenaventura

DESCRIPCIÓN

Objeto portátil seguro

ÁMBITO DE LA INVENCIÓN

Esta invención se refiere al área de objetos portátiles seguros generalmente estandarizados tales como tarjetas inteligentes. Dichos objetos comprenden (a) un cuerpo de tarjeta y (b) un micromódulo. El micro-módulo comprende un procesador y una o más memorias en las que se almacenan una primera y una segunda aplicación.

TÉCNICA ANTERIOR

Las tarjetas inteligentes son generalmente objetos portátiles seguros estandarizados con recursos de hardware y software particularmente limitados.

Algunas de las tarjetas actualmente disponibles, particularmente las denominadas tarjetas Java Card™, comprenden una máquina virtual. Dicha máquina virtual comprende un motor de ejecución (intérprete). El motor de ejecución es capaz de interpretar instrucciones de una aplicación de tarjeta inteligente y ejecutarlas en un espacio de ejecución que en la práctica está definido por dicha máquina virtual.

Algunas tarjetas implementan una pluralidad de aplicaciones Java Card™. En ese caso, todas las aplicaciones Java Card™ se ejecutan dentro del mismo espacio de ejecución y, a menudo, se rigen por las mismas reglas de seguridad. Así, aunque eso no sea necesario, algunas aplicaciones que no requerirían un alto nivel de seguridad para su ejecución, pueden acceder a recursos confidenciales a los que no deberían poder acceder, dentro del espacio de ejecución definido por la máquina virtual. Eso lleva a un problema de seguridad. Además, e incluso si eso es necesario solo para algunas aplicaciones, se implementan amplias medidas de seguridad para todas las aplicaciones, incluidas aquellas en las que la aplicación de tales medidas no está justificada en vista de la confiabilidad esperada. En un ejemplo, la ejecución de cálculos redundantes o el almacenamiento redundante de datos se usa para todas las aplicaciones, independientemente de lo que sean. Como resultado, los recursos de la tarjeta se utilizan para la ejecución de aplicaciones que no los necesitan, con un efecto en el rendimiento de la tarjeta. La Solicitud de Patente Internacional WO/0137085 A1 revela un método para la carga de aplicaciones en una plataforma multi-aplicación.

La Solicitud de Patente Europea RP 1 909 244 A1 revela un dispositivo de ejecución que comprende dos unidades de ejecución, una unidad de ejecución normal y una unidad de ejecución protegida.

RESUMEN DE LA INVENCIÓN

En vista de lo anterior, un problema técnico que la invención se propone resolver es el de hacer objetos portátiles seguros del tipo de tarjeta inteligente que puede ejecutar una pluralidad de aplicaciones, compensando los inconvenientes antes mencionados de la técnica anterior, donde la seguridad de ejecución de dicha pluralidad de aplicaciones se refuerza, particularmente cuando estas aplicaciones no requieren acceso a ciertos recursos de seguridad, y donde el impacto de la implementación de medidas de seguridad en el rendimiento de la tarjeta es limitado, cuando dichas medidas no están justificadas para la ejecución de algunas aplicaciones.

La solución proporcionada por la invención a este problema técnico comprende, en primer lugar, un objeto portátil seguro del tipo de tarjeta inteligente, de acuerdo con la reivindicación 1. En segundo lugar, comprende un método para asegurar un objeto portátil seguro del tipo de tarjeta inteligente, de acuerdo con la reivindicación 12. Así, si la primera aplicación, por ejemplo, se debe ejecutar en un espacio de ejecución que se extiende a recursos de seguridad críticos, la segunda aplicación no podría tener acceso a dichos recursos, lo que hace que el objeto sea más seguro. Además, será posible implementar medidas de seguridad diferentes de una aplicación a otra, dependiendo de los motores de ejecución, mejorando así el rendimiento de la tarjeta.

Ventajosamente, (a) el segundo motor de ejecución es distinto del primer motor de ejecución; (b) el primer motor de ejecución es el motor de ejecución de una primera máquina virtual y el segundo motor de ejecución es el motor de ejecución de una segunda máquina virtual; (c) las máquinas virtuales son máquinas virtuales Java Card™ y la primera y segunda aplicaciones son aplicaciones Java Card™; (d) las medidas de seguridad requeridas por las máquinas virtuales para la ejecución de las aplicaciones que administran en su espacio de ejecución son diferentes de una máquina virtual a otra, y las aplicaciones se organizan en dichas máquinas virtuales dependiendo del nivel de seguridad requerido para su ejecución; (d) las medidas de seguridad requeridas por las máquinas virtuales para ejecutar las aplicaciones que gestionan en su espacio de ejecución son diferentes de una máquina virtual a otra, y las aplicaciones se organizan en dichas máquinas virtuales según su certificación o el resultado de su verificación; (e) un espacio de ejecución se extiende a un recurso específico y otro espacio de ejecución no se extiende a dicho recurso específico; (f) el recurso específico es una zona de memoria y los datos almacenados en dicha zona de memoria, un canal de comunicación, dispositivos de comunicación o almacenamiento de datos o una biblioteca de códigos; (g) los recursos de las máquinas virtuales son compartidos; (h) los recursos compartidos incluyen la

memoria dedicada a las aplicaciones o algunas bibliotecas de códigos comunes a las máquinas virtuales; (i) dicho objeto es un módulo de identificación de abonado diseñado para insertarse en un teléfono móvil que comprende un módulo para comunicación de radiofrecuencia de campo cercano, dicho módulo de identificación de abonado tiene una primera interfaz de comunicación encaminada a dicho módulo para comunicación de radiofrecuencia de campo cercano y segunda interfaz de comunicación, y el acceso a la primera interfaz de comunicación solo está permitido en el primer espacio de ejecución; (j) la primera aplicación es ejecutada exclusivamente por el primer motor de ejecución en el primer espacio de ejecución, con exclusión de cualquier ejecución por el segundo motor de ejecución en el segundo espacio de ejecución, y la segunda aplicación es ejecutada exclusivamente por el segundo motor de ejecución en el segundo espacio de ejecución, con exclusión de cualquier ejecución por parte del primer motor de ejecución en el segundo espacio de ejecución; (k) el método según la invención comprende además etapas durante las cuales: (a) se identifica el espacio de ejecución para asignar a cada aplicación a instalar y dicha aplicación se instala para que se ejecute en dicho espacio de ejecución; (b) el espacio de ejecución está identificado al menos por la identidad del proveedor de dicha aplicación; (c) el espacio de ejecución se identifica al menos por el nivel de verificación o certificación de dicha aplicación; (d) dicho primer espacio de ejecución proporciona acceso a un recurso, dicho segundo espacio de ejecución no permite el acceso a dicho recurso, y dicho espacio de ejecución para asignar se identifica al menos por la necesidad de dicha aplicación para acceder a dicho recurso; y (e) dicho recurso es una interfaz de comunicación de dicho objeto portátil seguro.

BREVE DESCRIPCIÓN DE FIGURAS Y DIBUJOS

La invención se comprenderá mejor después de leer la descripción no limitativa que sigue a continuación, que se refiere a los dibujos adjuntos, donde:

- la figura 1 es una ilustración esquemática de un ejemplo de realización de un objeto de acuerdo con la invención, donde dicho objeto comprende dos máquinas virtuales Java Card™ separadas;
- la figura 2 es una ilustración esquemática de un ejemplo de realización de un objeto de acuerdo con la invención, donde dicho objeto comprende dos máquinas virtuales, la memoria dedicada a las aplicaciones de las cuales se pone en común; y
- la figura 3 es una ilustración esquemática de un ejemplo de realización de un objeto de acuerdo con la invención, donde dicho objeto comprende dos máquinas virtuales, y donde una primera máquina virtual accede al primer medio de comunicación y una segunda máquina virtual accede a un segundo medio de comunicación, estando dicho primer y segundo medio de comunicación separados.

DESCRIPCIÓN DETALLADA DE LA INVENCIÓN

La invención se refiere a un objeto portátil seguro del tipo de tarjeta inteligente. El objeto comprende, primeramente, un cuerpo de objeto, por ejemplo un cuerpo de tarjeta de plástico de dimensiones estandarizadas y, en segundo lugar, un micro-módulo.

El micro-módulo del objeto de acuerdo con la invención comprende al menos un procesador y al menos una memoria en la que se almacenan al menos dos aplicaciones y sus datos. Las dos aplicaciones son ejecutadas intermitentemente por el procesador.

Los recursos de los objetos de acuerdo con la invención, y particularmente los recursos de hardware y más particularmente el tamaño de la memoria, son limitados.

De acuerdo con la invención, una primera aplicación es ejecutada por un primer motor de ejecución en un primer espacio de ejecución y una segunda aplicación es ejecutada por un segundo motor de ejecución en un segundo espacio de ejecución. Ventajosamente, el primer motor de ejecución es parte de una primera máquina virtual y el segundo motor de ejecución es parte de una segunda máquina virtual. El objeto de acuerdo con la invención incluye así al menos dos máquinas virtuales, p. Máquinas virtuales Java Card™. Las dos máquinas virtuales posiblemente pueden compartir algunos recursos. Las máquinas virtuales pueden proporcionar diferentes niveles de seguridad.

Espacios de ejecución

Cada aplicación está diseñada para ejecutarse en un espacio de ejecución con ciertas características. Estas características pueden incluir lo siguiente, por ejemplo:

- la naturaleza del espacio de ejecución y, más específicamente, el tipo de procesador o máquina virtual definida por un cierto conjunto de instrucciones;
- la provisión de medios para acceder a los recursos, como la posibilidad de leer o editar algunos datos en la memoria, en particular de acuerdo con el grado de confidencialidad de los datos, o su naturaleza crítica (importancia de proteger su integridad), canales de comunicación con dispositivos periféricos u otros sistemas, o bibliotecas de códigos;
- medidas de seguridad destinadas a proteger el código de aplicaciones y los datos manejados por ellos desde otras aplicaciones o partes externas.

La invención se refiere a objetos que comprenden al menos dos aplicaciones diseñadas para ejecutarse en espacios de ejecución con al menos la naturaleza del espacio de ejecución como característica común. Por ejemplo, pueden ser dos aplicaciones Java Card™, ambas diseñadas para ejecutarse en una máquina virtual Java Card™. En cuanto a sus otras características, algunas son comunes y otras son diferentes.

De acuerdo con la invención, el objeto proporciona las dos aplicaciones con espacios de ejecución separados. Sin embargo, estos espacios comparten algunos componentes, según los requisitos comunes a las dos aplicaciones.

De manera más general, el objeto según la invención puede proporcionar varios espacios de ejecución y puede permitir la ejecución de varias aplicaciones, ya que cada aplicación tiene su propio espacio de ejecución o con las aplicaciones agrupadas en un número menor de espacios de ejecución de acuerdo con criterios tales como el conjuntos de características requeridas y el nivel requerido de seguridad.

Uso compartido de recursos

El objeto portátil comprende diferentes recursos en cantidad limitada, que se distribuyen entre las aplicaciones. Un recurso de dicho objeto es la memoria. La mayoría de los objetos comprenden además componentes funcionales adicionales que permiten la interacción con los usuarios, el intercambio de datos con otros sistemas informáticos o la acción u observación de sistemas físicos externos. Todos estos recursos deben distribuirse entre las aplicaciones, ya sea porque un recurso dado se ha reservado para una aplicación, o porque se comparte entre varias aplicaciones, en cuyo caso el objeto de acuerdo con la invención administra esta distribución.

Memoria

El objeto de acuerdo con la invención comprende una o más memorias, que pueden ser de uno o más tipos. La memoria puede ser, por ejemplo, una RAM, una memoria permanente regrabable, como del tipo EEPROM o una ROM. La cantidad de memoria contenida en los objetos de acuerdo con la invención es particularmente limitada. En particular, el tamaño de la memoria ROM es de unos pocos cientos Kb y el tamaño de la memoria EEPROM es de unas pocas decenas de Kb.

Cada aplicación es un programa de computadora, cuyo código se almacena en una memoria del objeto portátil de acuerdo con la invención. Además, cada aplicación almacena datos en al menos una memoria.

De acuerdo con la invención, cada espacio de ejecución tiene una o más porciones de memoria que se le asignan específicamente. Otras porciones de memoria pueden asignarse adicionalmente a componentes de software que se comparten entre varios espacios de ejecución, o pueden formar zonas de memoria compartidas que son accesibles simultáneamente para varios espacios de ejecución.

Recursos reservados

Algunos recursos están reservados para una aplicación o una categoría de aplicaciones. Los recursos son gestionados directamente por un espacio de ejecución y cualquier aplicación que necesite usar el recurso es ejecutada por ese espacio de ejecución.

Algunos objetos tienen uno o más dispositivos de comunicación o almacenamiento de datos que algunas aplicaciones solo pueden usar legítimamente. En ese caso, es ventajoso administrar estos dispositivos directamente dentro de un espacio de ejecución. Las aplicaciones que necesitan acceder a los dispositivos se ejecutan dentro de dicho espacio de ejecución, mientras que las otras aplicaciones se ejecutan en espacios de ejecución que no ofrecen dicho acceso.

Un objeto de acuerdo con la invención puede comprender recursos que solo pueden ser utilizados por proveedores aprobados. Estos recursos pueden ser dispositivos periféricos, pero también, por ejemplo, bibliotecas de códigos, cuyo uso requiere el pago de una tarifa. En ese caso, es ventajoso ejecutar las aplicaciones aprobadas en un espacio de ejecución con los recursos afectados, y las otras aplicaciones en un espacio de ejecución que no permite el acceso a dichos recursos.

Recursos compartidos

Algunos recursos deben ser compartidos entre varias aplicaciones, o incluso entre todas las aplicaciones. Resulta entonces ventajoso autorizar y permitir que los diferentes espacios de ejecución usen dichos recursos. Una manera de lograr eso es administrar cada recurso involucrado con la ayuda de un componente de software específico ubicado lógicamente fuera de los espacios de ejecución. Los espacios de ejecución usan el componente para acceder al recurso en cuestión.

Canales de comunicación

Un caso específico importante de un recurso potencialmente compartido es un medio de comunicación. Si solo una aplicación usa ese medio de comunicación, es posible hacer que los medios de comunicación sean gestionados por un espacio de ejecución y asegurarse de que la aplicación sea ejecutada por ese espacio de ejecución. Por otro lado, si es probable que varias aplicaciones utilicen los medios de comunicación, un componente de software debe permitir compartir los medios de comunicación. El componente puede ser parte de un espacio de ejecución, en cuyo caso solo las aplicaciones que se ejecutan en dicho espacio de ejecución tienen acceso a dichos medios de comunicación, o están fuera de los espacios de ejecución y se pueden usar por todos los espacios de ejecución para que todas las aplicaciones pueden acceder a dichos medios de comunicación.

En un modo de realización de la invención, las aplicaciones que pueden ejecutarse en el objeto de acuerdo con la invención se indican mediante un identificador, por ejemplo, un número o una cadena de caracteres, y los medios de comunicación permiten recibir mensajes donde el destinatario se indica en forma de identificador de la aplicación que manejará el mensaje. En ese caso, el componente de software responsable del procesamiento de primera línea de los mensajes recibidos decodifica al menos parcialmente cada mensaje recibido para identificar el identificador de la aplicación que es el destinatario del mensaje. Dicho componente de software puede acceder además a una tabla de correspondencia que asocia, para cada aplicación que probablemente reciba mensajes, el identificador de dicha aplicación con el identificador del espacio de ejecución en el que se ejecuta dicha aplicación. El mencionado componente de software luego reenvía el mensaje al espacio de ejecución en el que se ejecuta la aplicación del destinatario. Si resulta necesario, dicho componente de software activa el procesamiento del mensaje por parte de la aplicación receptora, por ejemplo, activando su espacio de ejecución y activando el procedimiento para gestionar los mensajes recibidos desde dicha aplicación.

Distribución de componentes de software

El objeto portátil seguro según la invención comprende al menos dos espacios de ejecución seguros del mismo tipo. Estos espacios están definidos por máquinas virtuales. Estas pueden ser, por ejemplo, máquinas virtuales Java Card™. Un espacio de ejecución generalmente se define por varios componentes de software, entre los cuales se puede identificar un motor de ejecución formado por la máquina virtual *stricto sensu*, donde los otros componentes aparecen como elementos adicionales como controladores de dispositivos, administradores de memoria, administradores de seguridad o bibliotecas de códigos (API). En la invención, algunos de estos componentes se comparten entre los espacios de ejecución. En otras palabras, los códigos de dichos componentes solo se almacenan una vez en la memoria. Por su parte, otros componentes son específicos de un espacio de ejecución. De esta forma, los otros componentes solo son accesibles a un espacio de ejecución para la exclusión de otro, o existen varias versiones de estos otros componentes, pero con diferentes propiedades.

Motor de ejecución

En la invención, los espacios de ejecución son de la misma naturaleza, es decir, el formato de aplicación que pueden ejecutar es el mismo. En particular, es natural, siempre que sea posible, utilizar el mismo motor de ejecución para diferentes aplicaciones y compartir así el motor de ejecución. Si las aplicaciones están en formato nativo, es decir, destinadas a ser ejecutadas directamente por un procesador (componente físico), el motor de ejecución es dicho procesador, que se comparte intrínsecamente. Si las aplicaciones están en otro formato, es decir, si un motor de ejecución adaptado es una máquina virtual, es posible compartir el código que implementa la máquina virtual, pero no se impone.

En un modo de realización de la invención, incluso si los motores de ejecución tienen las mismas propiedades funcionales fundamentales porque son capaces de ejecutar las mismas aplicaciones, los motores de ejecución tienen diferentes características secundarias, y por lo tanto tienen códigos diferentes. Por ejemplo, se han presentado ejemplos de características secundarias que un motor de ejecución puede tener, pero no el otro.

En un ejemplo de realización, un motor de ejecución comprende medidas de seguridad adicionales diseñadas para proteger los datos manejados por la aplicación. Por ejemplo, los datos almacenados en el objeto o comunicados con un sistema externo pueden estar encriptados. Dado que estas medidas de seguridad tienen un costo en términos de tiempo de cálculo y posiblemente de tamaño de memoria, otro motor de ejecución no incluye dichas medidas de seguridad para ofrecer un mejor rendimiento a las aplicaciones que no manejan datos confidenciales.

En un ejemplo de realización, un motor de ejecución comprende medidas de seguridad adicionales diseñadas para proteger la integridad de los datos gestionados, incluso si la ejecución se interrumpe, por ejemplo, si el suministro de energía al objeto se interrumpe repentinamente o si dicho objeto se sobrecalienta o si un componente físico del objeto está dañado. Las perturbaciones previstas también pueden incluir ataques al objeto por parte de un tercero que intenta leer datos a los que no debería tener acceso o lograr un comportamiento ilegítimo por parte del objeto, por ejemplo para autorizar una transacción bancaria que no debería autorizarse. Dichas medidas de seguridad a menudo incluyen la ejecución de cálculos redundantes o el almacenamiento de datos redundantes. Por lo tanto, tienen un coste que no está necesariamente justificado en vista de la confiabilidad esperada. Otro motor de

ejecución no proporciona dicha redundancia, que también ofrece un mejor rendimiento en detrimento de la integridad de los datos. De acuerdo con la invención, un motor de ejecución incluye el control de accesos de memoria por las aplicaciones ejecutadas por él para evitar accesos fuera de las porciones de memoria que están asignadas a la aplicación, mientras que otro motor de ejecución no proporciona tales controles. El segundo motor de ejecución, que ejecuta aplicaciones más rápido y con menores requisitos de memoria, se destina a aplicaciones en las que se supone que el proveedor es lo suficientemente confiable, que puede reconocerse mediante un certificado que acompañe a la aplicación o las aplicaciones en las que los primeros verificado de forma automática o manual.

En un ejemplo de realización, un motor de ejecución es más rápido que el otro, pero a un costo mayor en términos de algunos recursos físicos, como el consumo de energía. En este caso, la elección del motor de ejecución es el resultado de un compromiso entre el rendimiento de ejecución y el consumo de recursos físicos.

En un ejemplo de realización, un motor de ejecución es más eficiente que el otro, pero su uso está sujeto a un precio que depende del número de aplicaciones ejecutadas por él o de características tales como el tamaño o la complejidad de las aplicaciones.

En una variante del modo de la invención que se acaba de describir, los espacios de ejecución son del mismo tipo pero diferentes en algunos detalles. Los espacios de ejecución pueden, por ejemplo, ser definidos por máquinas virtuales que cumplen el mismo estándar, pero adicionalmente proporcionan algunas extensiones que no son necesariamente compatibles entre sí. Resulta entonces ventajoso el proporcionar a los motores de ejecución un espacio de ejecución para cada familia de extensiones incompatible.

En otra variante de la invención, los motores de ejecución comparten parte del código. Eso es posible, e incluso deseable, siempre que los motores de ejecución estén lo suficientemente parecidos, y solo difieran, por ejemplo, en la realización de ciertas operaciones particulares. Varios modos de realización de esta variante son posibles. En dos modos de realización, los motores de ejecución son máquinas virtuales donde una parte del código, llamada descodificación, ejecutada para cada instrucción, tiene como objetivo decodificar la operación llevada a cabo por la instrucción y transferir el control a otra parte del código, que es específico para cada operación.

En un primer modo de realización, el núcleo del motor de ejecución, que comprende particularmente el código de decodificación de instrucciones, es común a los dos espacios de ejecución. Una celda de memoria contiene un identificador del espacio de ejecución activo. Cuando el código de descodificación detecta una instrucción que debe ejecutarse de manera diferente según el espacio de ejecución, lee la dirección del código correspondiente a la operación que se llevará a cabo en el espacio de ejecución activo desde una tabla, y transfiere el control al código ubicado de esta manera.

En un segundo modo de realización, cada espacio de ejecución tiene un motor de ejecución que es específico para él. Cuando el código de decodificación detecta una instrucción que debe ejecutarse de la misma manera en ambos espacios de ejecución considerados, dicho código de decodificación transfiere la ejecución a una parte del código que es común a los dos espacios de ejecución.

Debe observarse que las máquinas virtuales pueden estar contenidas en la misma memoria o en diferentes memorias del objeto de acuerdo con la invención. En el primer ejemplo, dos máquinas virtuales Java Card™ están contenidas en una sola memoria ROM del objeto. En un segundo ejemplo, una primera máquina virtual está contenida en una memoria ROM de dicho objeto, y una segunda máquina virtual está contenida en una memoria diferente del objeto, en una memoria EEPROM.

Componentes adicionales

Con independencia del hecho de que los espacios de ejecución comparten el motor de ejecución o de otro modo, pueden compartir o tener sus propios componentes adicionales diferentes. Por ejemplo, una máquina virtual Java Card™ típica comprende, además del motor de ejecución responsable de interpretar las instrucciones que componen el código de las aplicaciones, bibliotecas diferenciadas como Java Card API (interfaz de programación de aplicaciones), un cortafuegos, un registro de aplicaciones, un administrador de memoria, un despachador y una serie de bibliotecas desarrolladas adicionalmente según las necesidades de las aplicaciones. Dependiendo del caso, puede ser preferible que cada componente sea compartido por las diferentes aplicaciones para que cada aplicación tenga su propio componente. Normalmente, compartir permite ahorrar memoria y facilita la comunicación entre aplicaciones, mientras que la separación aumenta el aislamiento y permite diferentes implementaciones del componente en cuestión. Cabe señalar que en cada caso, algunos componentes se pueden compartir, mientras que otros son separados.

Elección del espacio de ejecución

En la invención, a cada aplicación se le asigna un espacio de ejecución desde los espacios de ejecución disponibles en el objeto. Son posibles varios modos de realización de esta asignación, que difieren particularmente en el momento en que se identifica la asignación y cuando se hace efectiva. Además, la asignación puede basarse en

varios criterios. Aquí se mencionan algunos ejemplos de métodos de asignación y criterios de asignación, entendiéndose que son posibles otros modos.

Métodos de asignación

La instalación de una aplicación en un objeto portátil seguro de acuerdo con la invención implica al menos cargar el código de la aplicación en el objeto, de modo que dicho código se coloca en una memoria del objeto. La carga puede consistir en transmitir dicho código al objeto después de programar adecuadamente el objeto para almacenar dicho código en dicha memoria, pero son posibles otros métodos, particularmente el código de la aplicación puede colocarse en una ROM del objeto cuando dicho objeto está hecho. En la invención, el método para instalar la aplicación implica identificar el espacio de ejecución asignado a la aplicación. El espacio de asignación se puede identificar en uno o más pasos, antes, después o junto con la carga.

En un primer modo de asignación, cuando el código de una aplicación se carga en una memoria del objeto, el código se coloca en una zona de esa memoria que está asignada a uno de los espacios de ejecución, identificados de acuerdo con los criterios de asignación. La memoria puede ser, por ejemplo, una ROM si dicho código se carga mientras se fabrica el objeto, o una memoria regrabable del tipo EEPROM si el código se carga después de que se haya fabricado el objeto. La aplicación se asigna a ese espacio de ejecución, que se encarga de instalarlo y ejecutarlo.

En un segundo modo de asignación, cuando el código de la aplicación se carga en una memoria del objeto, tiene lugar un proceso de instalación que implica la ejecución del código suministrado con la aplicación, un código que ya está presente en el objeto o una combinación de los dos. Durante este método, el espacio de ejecución se identifica sobre la base de criterios de asignación, y la información que asocia la aplicación con el espacio de ejecución identificado se almacena en la memoria. Opcionalmente, el espacio de ejecución ejecuta un procedimiento adicional para instalar la aplicación. Cuando la aplicación necesita ser ejecutada, el componente de software responsable de la ejecución lee dicha información asociando la aplicación con el espacio de ejecución y desencadena la ejecución de la aplicación en el espacio de ejecución identificado de esa manera.

En un tercer modo de asignación, cuando se ejecuta la aplicación, ya sea a petición explícita del usuario del objeto o siguiendo un estímulo interno o externo como un evento de tiempo (por ejemplo, un evento de temporizador o un evento de tiempo de espera) o la recepción de una solicitud en un dispositivo o un medio de comunicación, el componente de software responsable de la ejecución identifica el espacio de ejecución sobre la base de los criterios de asignación y desencadena la ejecución de la aplicación por el espacio de ejecución identificado de esa manera.

Criterios de asignación

La asignación puede ser determinada por el desarrollador o el proveedor de la aplicación. En ese caso, la aplicación, tal como ha sido distribuida por su desarrollador o proveedor y cargada en el objeto, incluye, además de su código ejecutable, al menos una información que identifica el espacio de ejecución a usar.

Es posible asignar un espacio de ejecución diferente a cada desarrollador o proveedor de aplicaciones o grupos de desarrolladores o proveedores. En ese caso, la aplicación se carga en la tarjeta con una información que identifica al desarrollador o proveedor de la tarjeta.

El espacio de ejecución puede ser seleccionado por la persona que instale la aplicación en el objeto o automáticamente en el objeto durante o después de la instalación, dependiendo de las indicaciones suministradas con la aplicación. La información se relaciona con las necesidades de la aplicación o las recomendaciones relacionadas con su ejecución. La información puede, por ejemplo, referirse a:

- el nivel de seguridad esperado por la aplicación;
- el nivel de confidencialidad de los datos generalmente manejados;
- el nivel de confiabilidad esperado por la aplicación;
- los dispositivos que la aplicación probablemente usará; los medios de comunicación utilizados por la aplicación;
- las extensiones a ser soportadas por el motor de ejecución;
- las bibliotecas de códigos requeridas para que la aplicación funcione;
- más generalmente, cualquier recurso requerido para que la aplicación funcione.

Por ejemplo, si un primer espacio de ejecución permite el acceso a un determinado dispositivo mientras que un segundo espacio de ejecución niega el acceso al mismo dispositivo, las aplicaciones que solicitan acceso al dispositivo se asignarán al primer espacio de ejecución mientras que las otras aplicaciones se asignarán al segundo espacio de ejecución.

El espacio de ejecución puede determinarse mediante un análisis estático de la aplicación, es decir, mediante un examen manual o automático del código de la aplicación para determinar algunas características. Por ejemplo, puede:

- asignar un espacio de ejecución en el que algunos accesos de memoria no están protegidos del posible desbordamiento de las zonas de memoria autorizadas para las aplicaciones donde el análisis estático muestra que solo llevan a cabo accesos de memoria autorizados y asignan un espacio de ejecución en el que están todos los accesos de memoria verificado a aplicaciones donde el análisis estático muestra que podrían llevar a cabo accesos de memoria fuera de las zonas autorizadas; y/o
- asignar un espacio de ejecución que comprenda medidas para la protección de algunos ataques a aplicaciones donde el análisis estático muestre que son responsables de tales ataques, y asignar un espacio de ejecución sin tales medidas para la protección de las aplicaciones donde el análisis estático muestra que su comportamiento es no interrumpido por tales ataques.

Debe observarse que estos diferentes criterios se pueden combinar de diferentes maneras. Por ejemplo, los espacios de ejecución se pueden asignar a algunos proveedores de aplicaciones, y otras reglas se pueden usar para identificar el espacio de ejecución asignado a las aplicaciones de proveedores desconocidos. Se puede usar información que se proporciona opcionalmente con la aplicación, e identificar el espacio de ejecución mediante un análisis estático si falta la información. Por el contrario, un análisis estático, que no siempre es concluyente, se puede llevar a cabo en todos los casos, y las indicaciones suministradas con la aplicación solo se pueden usar cuando el análisis estático no es concluyente. Son posibles otros métodos de identificación que combinan los métodos de elección descritos aquí.

Ejemplo 1: máquinas virtuales separadas

En un primer ejemplo de realización de la invención ilustrada en la figura 1, el objeto portátil es una tarjeta inteligente en la que se ejecutan al menos dos máquinas virtuales Java Card™, VM 1 y VM 2.

Cada máquina virtual tiene su propio motor de ejecución Intérprete 1, Intérprete 2, sus propias bibliotecas JC API 1, JC API 2 y su propia memoria, o cúmulo, dedicada a las aplicaciones Cúmulo 1, Cúmulo 2. Esa organización ofrece la ventaja de ofrecer un alto nivel de aislamiento entre las aplicaciones ejecutadas en el primer espacio de ejecución y en el segundo espacio de ejecución, porque las máquinas virtuales VM 1 y VM 2 no permiten que las aplicaciones accedan a la memoria fuera de su cúmulo. Cúmulo 1, Cúmulo 2. Cada aplicación se almacena en el cúmulo de una máquina virtual particular, almacena estos datos en dicho montón y se ejecuta dentro de dicha máquina virtual.

Algunos componentes se comparten entre las máquinas virtuales. En una tarjeta inteligente convencional Java Card™ con una sola máquina virtual Java Card™, el registro es un componente de software que administra especialmente una lista de aplicaciones instaladas, con indicaciones para cada aplicación, como su disponibilidad (en instalación, ejecutable, bloqueado, etc.) y los permisos que se le han otorgado. En este ejemplo de realización, el registro se comparte entre los espacios de ejecución y contiene un identificador del espacio de ejecución asignado a la aplicación para cada aplicación, además de la información convencional.

Los controladores I/O de entrada/salida para los diferentes ISO 7816, SWP, USB, RF y otras interfaces de la tarjeta inteligente también se comparten entre las máquinas virtuales, por lo que la tarjeta solo tiene un controlador de entrada/salida para cada interfaz.

Los mensajes recibidos por la tarjeta inteligente son procesados por un componente de software que distribuye los mensajes analizando cada mensaje recibido para determinar la aplicación del destinatario y desencadena la ejecución de dicha aplicación en dicho mensaje. Ese despachador de mensajes recibidos también se comparte entre las máquinas virtuales. Cuando se recibe un mensaje, una vez que la aplicación del destinatario ha sido identificada, el despachador del mensaje recibido consulta el registro para identificar el espacio de ejecución en el que se ejecutará la aplicación del destinatario, y desencadena el procesamiento del mensaje por la aplicación dentro del mencionado espacio de ejecución.

Ejemplo 2: dominios de seguridad SD

Para distribuir las aplicaciones entre las máquinas virtuales, es ventajoso utilizar la infraestructura de clasificación existente, cuya implementación se describirá a continuación.

Como se ilustra en la figura 1, una aplicación particular llamada dominio de seguridad, SD1 y SD2 se ejecuta en cada una de las máquinas virtuales VM 1 y VM 2. El dominio de seguridad gestiona los datos relacionados con la seguridad de un conjunto de aplicaciones. En general, cada proveedor de aplicaciones también proporciona un dominio de seguridad que se utiliza para todas las aplicaciones del proveedor. El dominio de seguridad contiene especialmente las claves secretas específicas del proveedor y realiza algunas operaciones en nombre de otras aplicaciones, en particular la instalación de aplicaciones y la carga de datos, donde el código de las aplicaciones y los datos cargados se codifican y autentican para que solo el proveedor en cuestión (o un representante al que se le otorgan las claves requeridas) puede actuar en sus propias aplicaciones.

En el ejemplo aquí, es ventajoso que cada máquina virtual contenga solo un dominio de seguridad. De esta forma, cada máquina virtual VM 1, VM 2 está asociada a un proveedor de aplicaciones. De hecho, la separación de

máquinas virtuales proporciona un grado de aislamiento entre las aplicaciones que se ejecutan en la primera máquina virtual VM 1 y las aplicaciones que se ejecutan en la segunda máquina virtual VM 2. En particular, eso proporciona la seguridad de la imposibilidad de que las aplicaciones del proveedor 1 accedan a los datos del proveedor 2 y viceversa.

Cuando se carga una aplicación en la tarjeta, tiene un dominio de seguridad asociado, ya sea porque se supone que ya está cargada en la tarjeta y la aplicación hace referencia a ella o porque el dominio de seguridad se carga al mismo tiempo que la aplicación en cuestión. El administrador de aplicaciones de la tarjeta activa la instalación de la aplicación en la máquina virtual, lo que hace que el dominio de seguridad proporcionado funcione. Si aún no está presente en la tarjeta, el administrador de aplicaciones separa una parte de la memoria y comienza una nueva máquina virtual que usa esa parte de la memoria. Luego, el administrador de aplicaciones activa la instalación del nuevo dominio de seguridad en la nueva máquina virtual y luego la instalación de la aplicación recién cargada en la nueva máquina virtual. Si el dominio de seguridad no se proporciona ni está presente, o si el dominio de seguridad es nuevo y está provisto, pero no se puede iniciar una nueva máquina virtual (por ejemplo, porque la tarjeta no admite esa función o porque no hay suficiente cantidad de memoria disponible), de la aplicación falla.

Así, una tarjeta inteligente que cumpla con la implementación descrita aquí podría actuar de manera segura como el medio para aplicaciones con requisitos de alta seguridad. Por ejemplo, la tarjeta puede actuar simultáneamente como una tarjeta bancaria (con una máquina virtual (VM) dedicada a las aplicaciones proporcionadas por el banco del titular de la tarjeta) y como una tarjeta de transporte (con una máquina virtual dedicada a las aplicaciones proporcionadas por el operador de transporte). La tarjeta también podría ser una tarjeta SIM insertada en un teléfono móvil, y podría permitir el funcionamiento de una máquina virtual dedicada a aplicaciones de telecomunicaciones.

Ejemplo 3: diferentes máquinas virtuales

En este ejemplo, la tarjeta inteligente permite la ejecución de varias máquinas virtuales, que, aunque son capaces de ejecutar las mismas aplicaciones, tienen diferentes propiedades, por lo que para cada aplicación, algunas máquinas virtuales son más apropiadas que otras. Las diferencias se relacionan con la seguridad de las diferentes aplicaciones y los datos administrados por éstas.

Al cargar una aplicación en la tarjeta inteligente, el código de la aplicación se analiza mediante un componente de software denominado verificador. Este componente es cargado en la tarjeta y ejecutado o ejecutado antes de cargar *stricto sensu* por una entidad responsable de la seguridad de la tarjeta inteligente. El verificador ejecuta un análisis estático del código de la aplicación para determinar sus propiedades con implicaciones de seguridad. Su análisis puede ser respaldado por anotaciones opcionales suministradas con el código de la aplicación. En particular, el verificador puede concluir que no existe absolutamente ningún riesgo de que la aplicación pueda acceder a la memoria u otros recursos para los cuales no está autorizada. El verificador calcula así el nivel de seguridad de la aplicación, ya sea diferenciando las aplicaciones seguras (para las que garantiza la ausencia de accesos prohibidos) de las aplicaciones no seguras (donde no se ofrece ninguna garantía), o proporcionando un resultado más preciso.

Si el verificador identifica la aplicación como suficientemente segura, es cargada y posteriormente ejecutada por una máquina virtual que no verifica algunos accesos a la memoria oa otros recursos, a saber, los accesos que el verificador garantiza como imposibles. Si, por otro lado, el verificador no proporciona dicha garantía, la aplicación es ejecutada por una máquina virtual diferente que verifica cuidadosamente cada acceso potencialmente peligroso y rechaza los accesos no autorizados durante la ejecución. De esa manera, la misma tarjeta inteligente puede ejecutar cualquier aplicación adecuada, pero ejecuta aquellas aplicaciones que antes se puede garantizar que sean seguras.

En una alternativa de realización, la tarjeta inteligente proporciona dos máquinas virtuales capaces de ejecutar las mismas aplicaciones, pero la primera máquina virtual tiene medidas de protección contra ataques de software o hardware destinados a afectar la integridad o la confidencialidad de los datos manipulados por las aplicaciones que se ejecutan allí, mientras que la segunda máquina virtual no tiene tales medidas de protección. Las aplicaciones que manejan datos sensibles se asignan a la primera máquina virtual, mientras que las otras aplicaciones se asignan a la segunda, que se hace más eficiente por la ausencia de medidas de protección. La segunda máquina virtual también se puede usar para probar y depurar aplicaciones que están en desarrollo. Cabe señalar que en este último caso de uso, es necesario que ambas máquinas virtuales sean idénticas desde el punto de vista del comportamiento funcional de las aplicaciones que se ejecutan en él, pero diferentes desde el punto de vista de las observaciones externas.

Ejemplo 4: memoria compartida

En el ejemplo de realización presentado en la figura 2, la tarjeta inteligente comprende al menos dos máquinas virtuales Java Card™ VM1, VM2, donde cada máquina virtual tiene su propio motor de ejecución Intérprete 1, Intérprete 2 y sus propias bibliotecas JC API 1, JC API 2. Sin embargo, se comparte la memoria dedicada a las aplicaciones Cúmulo. Los componentes que son únicos por naturaleza, como el registro, el despachador de mensajes recibidos y los controladores IO para ISO 7816, SWP, USB RF, etc., son compartidos por igual por las diferentes máquinas virtuales.

El intercambio de la memoria dedicada a las aplicaciones permite guardar la memoria en relación con el caso en que cada máquina virtual tiene su propio espacio de memoria específico para las aplicaciones. De hecho, cada división de la memoria conduce a la pérdida de memoria, ya que una parte del espacio de memoria libre que se encuentra en una máquina virtual que no la necesita se pierde en la otra máquina virtual. Tal intercambio también puede permitir economías de escala en lo que respecta a la memoria dedicada a la gestión de la memoria. Este aspecto es particularmente importante para las tarjetas inteligentes donde los recursos, particularmente en términos de RAM, son limitados.

El hecho de que la memoria en la que se almacenan las aplicaciones se comparte también facilita el intercambio de datos entre las aplicaciones. Esa organización es, por lo tanto, preferible si se requiere la posibilidad de comunicación entre aplicaciones y si se puede proporcionar seguridad que no sea aislando las aplicaciones entre sí. Compartir la memoria también puede permitir que una aplicación llame a una interfaz proporcionada por otra aplicación que se ejecuta en otra máquina virtual y, por lo tanto, desencadenar, desde una máquina virtual, un código que se ejecuta en dicha otra máquina virtual.

Además, incluso si las máquinas virtuales están separadas, pueden tener acceso compartido a algunas bibliotecas de código que se implementan independientemente de la máquina virtual, p. código que maneja formatos de datos o bibliotecas de computación numérica.

Como parte de este ejemplo, las máquinas virtuales ofrecen diferentes posibilidades. Por ejemplo, una máquina virtual tiene medidas para protegerse de los ataques físicos destinados a eludir la ejecución de algunas verificaciones de seguridad mediante métodos tales como interrupciones en el suministro de energía de la tarjeta inteligente. La máquina virtual está dedicada a la ejecución de aplicaciones que manejan datos confidenciales, que el titular de la tarjeta inteligente podría intentar modificar de manera ilegítima. Otra máquina virtual no ofrece tales medidas de protección y, por lo tanto, es más rápida. Se usa para aplicaciones que no requieren un alto nivel de protección.

En otro ejemplo, las máquinas virtuales Java Card™ agregan diferentes extensiones, que pueden ser incompatibles con el lenguaje Java Card™. En ese caso, las aplicaciones que requieren una extensión específica para el idioma son ejecutadas por la máquina virtual que ofrece la extensión en cuestión. Las aplicaciones que no requieren ninguna extensión particular pueden ser ejecutadas por cualquier máquina virtual y la máquina virtual que ofrezca el mejor rendimiento se seleccionará preferiblemente.

Ejemplo 5: comunicaciones separadas

En este ejemplo, que se ilustra en la figura 3, la tarjeta inteligente es un módulo de identificación de abonado (tarjeta SIM) diseñado para insertarse en un teléfono móvil, donde dicho teléfono móvil comprende un módulo para comunicación de radiofrecuencia de campo cercano (módulo NFC).

La tarjeta tiene entonces una primera interfaz lógica para la comunicación con el teléfono y, más específicamente, con el módulo NFC de dicho teléfono. Esa primera interfaz lógica es una interfaz que implementa la comunicación de la tarjeta con el teléfono móvil utilizando el protocolo SWP (Single Wire Protocol). La interfaz se dirige entonces a dicho módulo y la comunicación se lleva a cabo utilizando el protocolo SWP.

Además, la tarjeta tiene una segunda interfaz lógica para la comunicación con el teléfono y, más específicamente, con el procesador principal de dicho teléfono. Esa segunda interfaz lógica es una interfaz que implementa la comunicación de la tarjeta con el teléfono sobre la base del protocolo o protocolos de comunicación definidos en la norma ISO 7816.

En este ejemplo, la tarjeta inteligente permite la ejecución de dos máquinas virtuales separadas Java Card™ VM 1 y VM 2. Cada una de las dos máquinas virtuales VM 1, VM 2 tiene su propio motor de ejecución Intérprete 1, Intérprete 2 y sus propias bibliotecas JC API 1, JC API 2. La primera máquina virtual (VM 1) está dedicada a aplicaciones sin contacto, por ejemplo, aplicaciones de transporte, y solo accede a la interfaz lógica para comunicarse con el módulo NFC del teléfono. La segunda máquina virtual VM 2 está dedicada a aplicaciones telefónicas y solo accede a la interfaz para comunicarse con el procesador del teléfono para implementar las funciones telefónicas del procesador.

Por tanto, el acceso a la primera interfaz de comunicación solo está autorizado en el primer espacio de ejecución, mientras que el acceso a la segunda interfaz de comunicación solo está autorizado en el segundo espacio de ejecución.

Debe observarse que algunos componentes permanecen ventajosamente compartidos entre las máquinas virtuales. Por lo tanto, la tarjeta solo tiene un registro de aplicaciones que enumera las aplicaciones instaladas en todas las máquinas virtuales. Por ejemplo, eso hace posible usar una interfaz para cargar una aplicación diseñada para comunicarse con una interfaz diferente.

Debe observarse además que la memoria dedicada a las aplicaciones Cúmulo 1, Cúmulo 2 puede o no compartirse entre las máquinas virtuales; los beneficios de cada enfoque se han discutido en relación con las implementaciones descritas anteriormente. Siempre se puede compartir una parte de la memoria, y particularmente las bibliotecas de códigos comunes a los dos espacios de ejecución, mientras se reserva, dentro de cada máquina virtual, memoria para los datos que es preferible no compartir, particularmente datos confidenciales o confidenciales.

REIVINDICACIONES

1. Un objeto portátil seguro del tipo de tarjeta inteligente que comprende

(a) un cuerpo de objeto y
(b) un micro-módulo que comprende un procesador y una o más memorias en las que se almacenan una primera y una segunda aplicación, **caracterizado porque**

comprende además una primer y segundo motor de ejecución capaces de ejecutar dicha primera y segunda aplicación, y **porque** dicha primera aplicación es ejecutada por dicho primer motor de ejecución en un primer espacio de ejecución y dicha segunda aplicación es ejecutada por dicho segundo motor de ejecución en un segundo espacio de ejecución distinto del primer espacio de ejecución, el primer y el segundo motor de ejecución, cada uno correspondiendo a un intérprete capaz de interpretar instrucciones de una aplicación de tarjeta inteligente y ejecutarlas en un espacio de ejecución, un espacio de ejecución que es definido por una máquina virtual, siendo el primer motor de ejecución distinto del segundo motor de ejecución que tiene una característica secundaria distinta del primer motor de ejecución, siendo la característica secundaria que el segundo motor de ejecución implementa medidas de seguridad adicionales destinadas a prevenir que sea accesible desde fuera una porción de memoria asignada a la segunda aplicación de una aplicación ejecutada por el segundo motor de ejecución.

2. Un objeto portátil de acuerdo con la reivindicación 1, **caracterizado porque** el primer motor de ejecución es un motor de ejecución (Intérprete 1) de una primera máquina virtual (VM 1) y **porque** el segundo motor de ejecución es un motor de ejecución (Intérprete 2) de una segunda máquina virtual (VM 2).

3. Un objeto portátil de acuerdo con la reivindicación 2, **caracterizado porque** las máquinas virtuales (VM 1, VM 2) son máquinas virtuales Java Card™ y porque la primera y la segunda aplicaciones son aplicaciones Java Card™.

4. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones 2 o 3, **caracterizado porque** las medidas de seguridad requeridas por las máquinas virtuales para la ejecución de las aplicaciones que gestionan en su espacio de ejecución son diferentes de una máquina virtual a otra, y **porque** las aplicaciones se organizan en dichas máquinas virtuales sobre la base del nivel de seguridad requerido para su ejecución.

5. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones 2 o 3, **caracterizado porque** las medidas de seguridad requeridas por las máquinas virtuales para la ejecución de las aplicaciones que gestionan en su espacio de ejecución son diferentes de una máquina virtual a otra, y **porque** las aplicaciones están dispuestos en dichas máquinas virtuales sobre la base. de su certificación o el resultado de su verificación.

6. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones precedentes, **caracterizado porque** uno del primero o segundo espacios de ejecución se extiende a un recurso específico y **porque** el otro de los espacios de ejecución no se extiende a dicho recurso específico.

7. Un objeto portátil de acuerdo con la reivindicación 6, **caracterizado porque** el recurso específico es una zona de memoria junto con los datos almacenados en dicha zona de memoria, un canal de comunicación, dispositivos de comunicación o almacenamiento de datos o una biblioteca de códigos.

8. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones 2 a 7, **caracterizado porque** se comparten los recursos de las máquinas virtuales (VM 1, VM 2).

9. Un objeto portátil de acuerdo con la reivindicación 8, **caracterizado porque** los recursos compartidos incluyen la memoria dedicada a las aplicaciones (Cúmulo) o a algunas bibliotecas de códigos comunes a las máquinas virtuales.

10. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones precedentes, **caracterizado porque** dicho objeto es un módulo de identificación de suscriptor diseñado para ser insertado en un teléfono móvil que comprende un módulo para comunicación de radiofrecuencia de campo cercano, y **porque** dicho módulo de identificación de suscriptor tiene una primera interfaz de comunicación encaminada hacia dicho módulo para comunicación de radiofrecuencia de campo cercano, y una segunda interfaz de comunicación, y **porque** el acceso a la primera interfaz de comunicación solo está autorizado en el primer espacio de ejecución.

11. Un objeto portátil de acuerdo con cualquiera de las reivindicaciones precedentes, **caracterizado porque** la primera aplicación es ejecutada exclusivamente por el primer motor de ejecución en el primer espacio de ejecución, con exclusión de cualquier ejecución por parte del segundo motor de ejecución en el segundo espacio de ejecución, la segunda aplicación es ejecutada exclusivamente por el segundo motor de ejecución en el segundo espacio de ejecución, con exclusión de cualquier ejecución realizada por el primer motor de ejecución en el segundo espacio de ejecución.

12. Un método para asegurar un objeto portátil seguro del tipo de tarjeta inteligente que comprende (a) un cuerpo

de objeto y (b) un micro-módulo que comprende un procesador y una o más memorias en las que se almacenan una primera y una segunda aplicación, **caracterizado porque** comprende las siguientes etapas, donde:

Se proporciona un primer y un segundo motor de ejecución capaces de ejecutar dichas primera y segunda aplicaciones; El primer motor de ejecución ejecuta la primera aplicación en un primer espacio de ejecución; y
El segundo motor de ejecución ejecuta la segunda aplicación en un segundo espacio de ejecución distinto del primer espacio de ejecución el primer y el segundo motor de ejecución, cada uno correspondiendo a un intérprete capaz de interpretar instrucciones de una aplicación de tarjeta inteligente y ejecutarlas en un espacio de ejecución, un espacio de ejecución que es definido por una máquina virtual, siendo el primer motor de ejecución distinto del segundo motor de ejecución que tiene una característica secundaria distinta del primer motor de ejecución, siendo la característica secundaria que el segundo motor de ejecución implementa medidas de seguridad adicionales destinadas a prevenir que sea accesible desde fuera una porción de memoria asignada a la segunda aplicación de una aplicación ejecutada por el segundo motor de ejecución.

13. Un método de acuerdo con la reivindicación 12, **caracterizado porque** comprende además etapas en las que:

Se identifica el espacio de ejecución que debe asignarse para cada aplicación a instalar; y
Dicha aplicación es instalada para que se ejecute en dicho espacio de ejecución.

14. Un método de acuerdo con la reivindicación 13, **caracterizado porque** el espacio de ejecución es identificado al menos por la identidad del proveedor de dicha aplicación.

15. Un método según la reivindicación 13, **caracterizado porque** el espacio de ejecución es determinado al menos por el nivel de verificación o certificación de dicha aplicación.

16. Un método de acuerdo con la reivindicación 13, **caracterizado porque** dicho primer espacio de ejecución permite el acceso a un recurso, dicho segundo espacio de ejecución no permite el acceso a dicho recurso, y dicho espacio de ejecución a asignar es identificado al menos por la necesidad de dicha aplicación de acceder a dicho recurso.

17. Un método de acuerdo con la reivindicación 16, **caracterizado porque** dicho recurso es una interfaz de comunicación de dicho objeto portátil seguro.

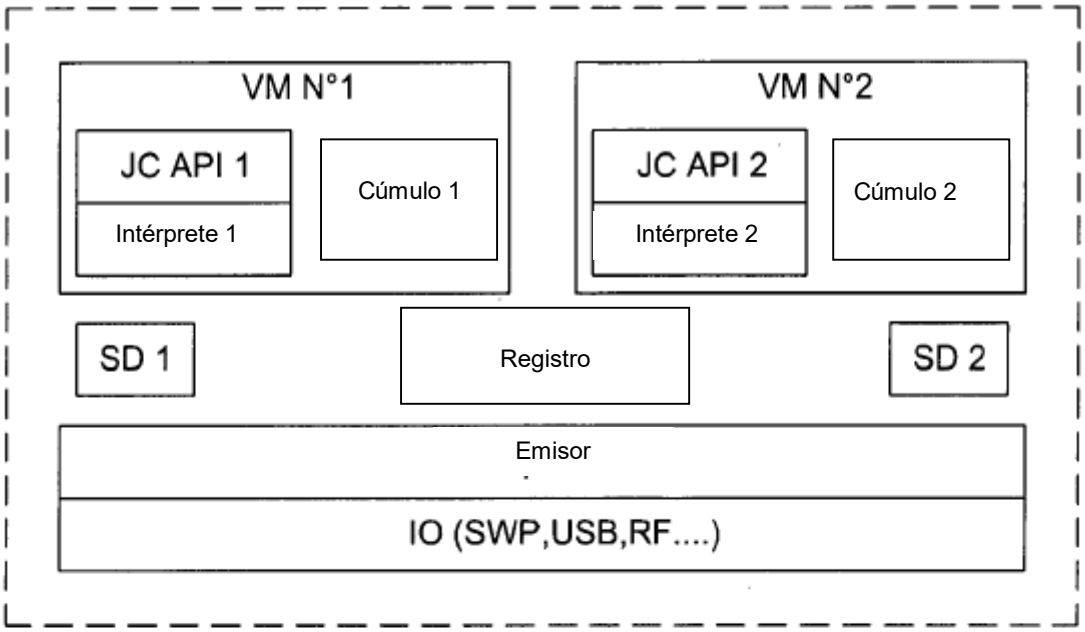
VM N°1
JC API 1
Intérprete 1
Cúmulo 1
VM N°2
JC API 2
Intérprete 2
Cúmulo 2
SD 1
Registro
SD 2
Emisor
IO (SWP,USB,RF....)

Fig. 1

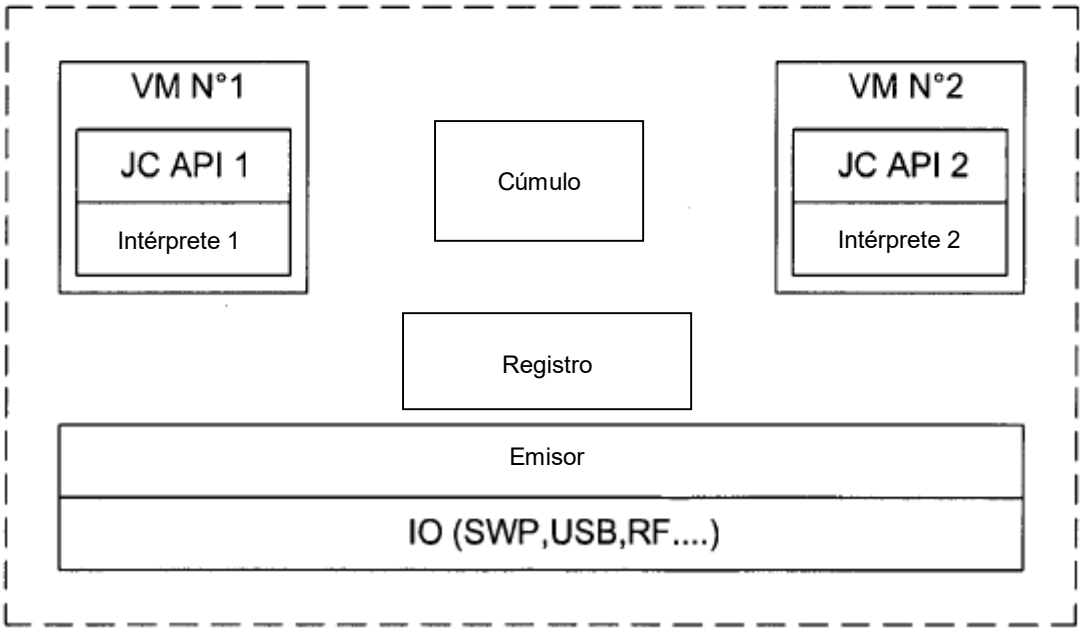
VM N°1
JC API 1
Intérprete 1
Cúmulo
VM N°2
JC API 2
Intérprete 2
Registro
Emisor
IO (SWP,USB,RF....)

Fig. 2

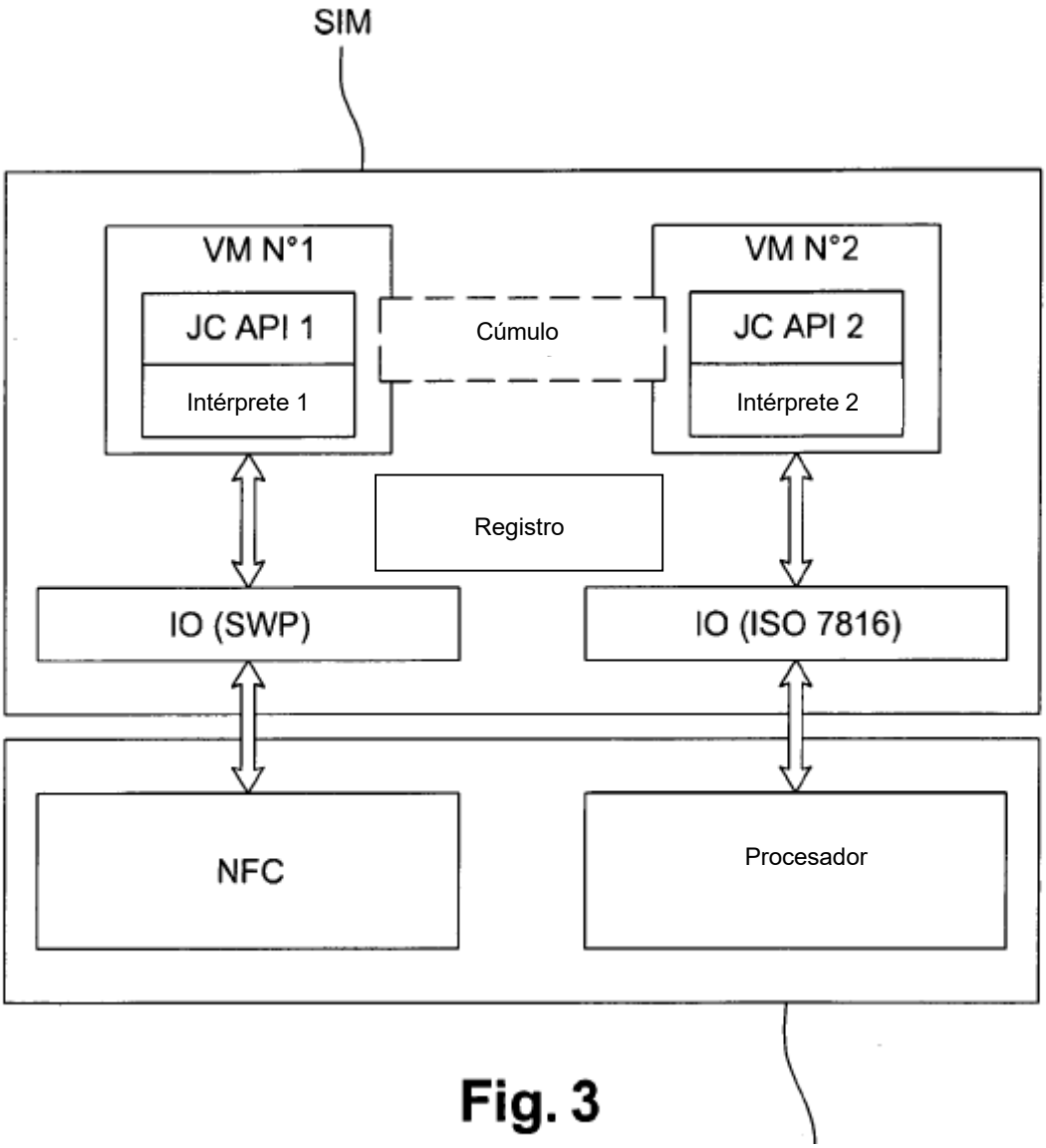
SIM
VM N°1
JC API 1
Intérprete 1
Cúmulo
VM N°2
JC API 2
Intérprete 2
Registro
IO (SWP)
IO (ISO 7816)
NFC
Procesador
Teléfono

Fig. 3