(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 734 305**

(51) Int. CI.:

G06F 11/00 (2006.01)
G06F 11/07 (2006.01)
G06F 11/34 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **22.12.2011 PCT/US2011/066821**

(87) Fecha y número de publicación internacional: **05.07.2012 WO12092124**

(96) Fecha de presentación y número de la solicitud europea: **22.12.2011 E 11853294 (4)**

(97) Fecha y número de publicación de la concesión europea: **01.05.2019 EP 2659371**

(54) Título: **Predicción, diagnóstico y recuperación de fallos de aplicaciones en base a patrones de acceso a recursos**

(30) Prioridad:

27.12.2010 US 978663

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
05.12.2019

(73) Titular/es:

MICROSOFT TECHNOLOGY LICENSING, LLC (100.0%)
One Microsoft Way
Redmond, WA 98052, US

(72) Inventor/es:

YOUNG, MATTHEW DAVID;
REIERSON, KRISTOFER HELLICK y
JEWART, ERIC

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Predicción, diagnóstico y recuperación de fallos de aplicaciones en base a patrones de acceso a recursos

Antecedentes

Las aplicaciones de software que se ejecutan en un sistema informático pueden fallar por una amplia variedad de razones, tales como errores de código, errores del usuario, datos de entrada incorrecta, recursos no disponibles, o similares. Dichos fallos de la aplicación pueden provocar la pérdida de datos y tiempo de inactividad de la aplicación, y pueden incurrir en costes y tiempo relacionados con la recuperación de la aplicación y los datos. Se espera que las aplicaciones que se ejecutan en un entorno común o desde una instalación común encuentren los mismos fallos dadas las mismas entradas, condiciones y/o circunstancias. Este puede ser el caso de las aplicaciones que se ejecutan en un entorno de aplicación virtualizado.

La virtualización de aplicaciones permite que las aplicaciones de software ejecutadas por un ordenador se desacoplen del hardware, el sistema operativo ("OS") y la configuración local del ordenador. La virtualización de aplicaciones puede eliminar el requisito de que una aplicación se instale, configure y mantenga localmente en el ordenador. En su lugar, un entorno de aplicación virtual puede ejecutarse en el ordenador y transmitir los componentes de la aplicación a través de una red desde un paquete de aplicación virtualizado que se mantiene centralmente en un servidor de aplicación virtual.

Es con respecto a estas consideraciones y otras que se presenta la divulgación realizada en la presente memoria.

El documento US2008250265 A1 describe un procedimiento para usar predicciones de fallo continuo para la gestión proactiva de fallos en sistemas de clústeres distribuidos que comprenden una pluralidad de nodos de ordenador principal. El procedimiento comprende la implementación de un sensor en cada nodo de ordenador principal, que recopila continuamente datos de registro sobre un ordenador principal local, y ejecuta operadores para generar flujos de registros. Los componentes de análisis, que residen en un nodo maestro, analizan continuamente las secuencias de registros recibidas y realizan predicciones de fallos utilizando procedimientos de clasificación basados en secuencias. Los diferentes tipos de fallos, como el agotamiento de los recursos que causa la pérdida de resultados de la consulta o la interrupción del procesamiento de la consulta, pueden describirse mediante predicados de fallo, que proporcionan los desarrolladores de la aplicación o los administradores del sistema. Para cada tipo de fallo, el componente de análisis mantiene un modelo de predicción de fallo que puede clasificar todos los valores de muestra asociados con la clase de fallo en tres estados: normal, previo al fallo y fallo. El modelo de predicción de fallos genera alarmas de fallo cuando las mediciones caen en el estado previo al fallo. Luego, el sistema realiza acciones preventivas basadas en el tipo de fallo pronosticado y el tipo de objeto que falla.

El documento WO2005008384 describe un procedimiento para identificar problemas en aplicaciones. El procedimiento comprende la supervisión, en un sistema de nivel de núcleo, el uso de recursos de una o más aplicaciones en ejecución sin modificar los entornos de tiempo de ejecución de las aplicaciones en ejecución. El recurso de supervisión comprende la determinación de los procesos que se ejecutan en un sistema y la obtención de estadísticas de recursos del sistema se obtiene para cada proceso en ejecución. Las estadísticas de recursos del sistema pueden obtenerse utilizando las instalaciones existentes, como el Q4, MDB, o cualquier otra utilidad de depuración del núcleo apropiada. Por ejemplo, al usar la utilidad Q4, se puede obtener y registrar el recuento de páginas de memoria asignado y utilizado por un proceso en ejecución. El recuento de páginas de memoria se puede ingresar en diferentes tipos de memoria que utiliza el proceso en ejecución: por ejemplo, tipos de texto, datos y pila. Desde el uso del sistema monitoreado, se puede identificar una aplicación cuyo patrón de uso del sistema cumpla con un criterio predeterminado asociado con uno o más problemas.

El documento WO2006077249 describe un procedimiento para reproducir, a partir de un archivo de registro, eventos internos dentro de un proceso que pertenece a una aplicación de software en una red. El procedimiento comprende ejecutar al menos una instrucción de repetición por el proceso de destino, iniciar una operación de repetición que devuelve, a este proceso de destino, al menos un dato de resultado obtenido por esta operación de repetición. El procedimiento comprende, además, realizar, mediante un agente de software externo al proceso de reinicio, un procedimiento denominado forzado que comprende tener en cuenta el proceso objetivo, en lugar del dato de resultados de la reproducción, de un dato forzado extraído de los datos de registro y que representa para la operación repetida un resultado dado denominado resultado registrado.

El objetivo de la presente invención es mejorar la predicción de las condiciones de error de un programa de aplicación.

Este objeto es resuelto por la materia objeto de las reivindicaciones independientes.

Las realizaciones se definen mediante las reivindicaciones dependientes.

En lo siguiente, cuando el término "realización" se refiere a combinaciones no reivindicadas de características, dicho término ha de entenderse como una referencia a ejemplos de la presente invención.

Las tecnologías se describen en la presente memoria para diferenciar el funcionamiento normal de un programa de aplicación de las condiciones de error para predecir, diagnosticar y recuperarse de los fallos de la aplicación. Cuando un programa de aplicación se ejecuta en un entorno de aplicación virtual, la capa de virtualización o el entorno de aplicación virtual puede tener conocimiento y control sobre las solicitudes que la aplicación realiza para los recursos, tales como lecturas de archivos de datos, escrituras en claves de registro y similares. Utilizando las tecnologías descritas en la presente memoria, la capa de virtualización puede registrar accesos a los recursos y, con el tiempo, establecer patrones comunes de uso de recursos. Una vez que se establecen dichos patrones de acceso a los recursos, la capa de virtualización puede continuar supervisando el uso de los recursos por parte del programa de la aplicación y proporcionar una advertencia o alerta cuando los patrones cambian. Esta advertencia proactiva puede proporcionar al usuario del programa de aplicación o al administrador la posibilidad de tomar acciones de diagnóstico o correctivas rápidamente, lo que reduce o incluso evita el tiempo de inactividad y la pérdida de datos.

Se apreciará que los programas de aplicación suelen tener un medio de mostrar errores al usuario o administrador, como un cuadro de diálogo emergente o un suceso registrado en un registro de sucesos de aplicación o sistema. Sin embargo, la calidad y la utilidad de estos mensajes de error pueden variar considerablemente de una aplicación a otra. Proporcionar mensajes de error de alta calidad que permitan determinar el origen del error requiere una inversión significativa, y no todos los proveedores de software realizan esta inversión. Aprovechar el registro de accesos a los recursos y los patrones comunes establecidos aquí descritos para correlacionar a qué recursos se estaba accediendo o cómo se cambió el patrón justo antes de que la condición de error permita a los usuarios y/o administradores diagnosticar más rápidamente los fallos de la aplicación e implementar acciones de recuperación, reduciendo el tiempo de inactividad de la aplicación.

Además, debido a la capa de virtualización es consciente de todo el uso de recursos por el programa de aplicación, se puede registrar incorporaciones de recursos, modificaciones o deleciones junto con los datos utilizados en estas modificaciones de recursos que se producen con el tiempo. Si el programa de la aplicación falla, se puede iniciar una segunda instancia del programa de la aplicación inmediatamente y se puede volver a reproducir el registro de las modificaciones de los recursos y los datos, restaurando así el estado de la aplicación a un punto justo antes del fallo de la primera instancia de la aplicación. Dicha conmutación por error rápida entre las instancias de la aplicación puede limitar el tiempo de inactividad adicional.

Según formas de realización, el acceso a los recursos por el programa de aplicación que se ejecuta en un entorno de aplicación virtual se supervisa, y los eventos de acceso de recursos se registran en un registro de acceso a los recursos. Los patrones de acceso a los recursos se establecen a partir de los eventos de acceso a recursos registrados utilizando técnicas de reconocimiento de patrones por ordenador. Si el acceso posterior a los recursos por parte del programa de la aplicación se desvía de los patrones establecidos, se notifica a un usuario y/o administrador del programa de la aplicación de una posible condición de error en función de la desviación detectada.

Además, las secuencias de eventos de acceso de recursos que se desvían de los recursos establecidos patrones de acceso pueden ser correlacionados con una condición de error basado en una proximidad temporal para el tiempo de ocurrencia de la condición de error para proporcionar información de diagnóstico para el usuario y/o administrador en cuanto al error. Finalmente, los eventos de acceso a los recursos relacionados con la adición, modificación o eliminación de los datos registrados en el registro de acceso a los recursos se pueden reproducir en el caso de un fallo de la aplicación y el posterior reinicio de la aplicación para restablecer el estado de la aplicación del programa de aplicación.

Se debe apreciar que la materia descrita anteriormente puede ser implementada como un aparato controlado por ordenador, un proceso de ordenador, un sistema informático, o como un artículo de fabricación tal como un medio legible por ordenador. Estas y otras varias características serán evidentes a partir de la lectura de la siguiente descripción detallada y una revisión de los dibujos asociados.

Este sumario se proporciona para introducir una selección de conceptos en una forma simplificada que se describen más adelante en la descripción detallada. Este sumario no tiene la intención de identificar características clave o características esenciales de la materia reivindicada, ni pretende ser utilizado como una ayuda para determinar el alcance de la materia reivindicada. Además, el objeto reivindicado no se limita a las implementaciones que resuelven cualquiera o todas las desventajas señaladas en cualquier parte de esta divulgación.

Breve descripción de los dibujos

La figura 1 es un diagrama de bloques que muestra aspectos de un entorno operativo ilustrativo y varios componentes de software proporcionados por las realizaciones presentadas en la presente memoria;
Las figuras 2-4 son diagramas de flujo que muestran procedimientos para diferenciar el funcionamiento normal de un programa de aplicación de las condiciones de error para predecir, diagnosticar y recuperarse de fallos de la aplicación, de acuerdo con las realizaciones descritas en la presente memoria; y
La figura 5 es un diagrama de bloques que muestra una arquitectura de software y hardware de ordenador ilustrativo para un sistema informático capaz de implementar aspectos de las realizaciones presentadas en la presente memoria.

Descripción detallada

La siguiente descripción detallada se dirige a las tecnologías para la diferenciación de la operación normal de un programa de aplicación de condiciones de error en base a los patrones de acceso a los recursos con el fin de predecir, diagnosticar, y/o recuperarse de errores de aplicación. Si bien el tema descrito aquí se presenta en el contexto general de los módulos de programa que se ejecutan junto con la ejecución de un sistema operativo y programas de aplicación en un sistema informático, los expertos en la técnica reconocerán que otras implementaciones pueden realizarse en combinación con otros tipos de módulos de programa. Generalmente, los módulos de programa incluyen rutinas, programas, objetos, artefactos físicos, estructuras de datos, etc. que realizan tareas particulares o implementan tipos de datos abstractos particulares. Además, los expertos en la materia apreciarán que la materia descrita en el presente documento se puede practicar con otras configuraciones de sistemas informáticos, incluidos dispositivos portátiles, sistemas multiprocesador, electrónica de consumo programable o basada en microprocesador, miniordenadores, ordenadores centrales y similares.

En la siguiente descripción detallada, se hace referencia a las figuras de los dibujos adjuntos que forman parte de la misma, y que muestran, a modo de ilustración, realizaciones específicas o ejemplos. En los dibujos que se acompañan, los números similares representan elementos similares a través de varias figuras.

La figura 1 muestra un entorno 100 operativo ilustrativo que incluye varios componentes de software para diferenciar el funcionamiento normal de un programa de aplicación de las condiciones de error para predecir, diagnosticar y recuperarse de fallos de la aplicación, de acuerdo con las realizaciones proporcionadas en el presente documento. El entorno 100 incluye un ordenador 102. El ordenador 102 puede ser un ordenador servidor; un ordenador personal ("PC"), como una estación de trabajo de escritorio, un ordenador portátil o una notebook; un asistente digital personal ("PDA"); un teléfono inalámbrico; un módulo de conexión; una consola de juegos; o cualquier otro dispositivo informático que pueda ejecutar programas de aplicación.

Un programa 104 de aplicación de software se ejecuta en el ordenador 102. De acuerdo con las realizaciones, el programa 104 de aplicación puede ejecutarse dentro de un entorno 106 de aplicación virtual. El entorno 106 de aplicación virtual puede permitir que el ordenador 102 inicie y ejecute programas de aplicación que no se hayan instalado previamente en el ordenador. El entorno 106 de aplicación virtual puede, en cambio, transmitir los componentes del programa 104 de aplicación en tiempo real o casi en tiempo real a través de una red 110 desde un servidor 112 de aplicación virtual. El entorno 106 de aplicación virtual y el servidor 112 de aplicación virtual pueden basarse en la tecnología MICROSOFT® APP-V de MICROSOFT Corporation de Redmond, Washington, la tecnología CITRIX XENAPP™ de CITRIX SYSTEMS Inc. de Fort Lauderdale, Florida, o cualquier otra aplicación de transmisión por secuencias. y plataforma o tecnologías de virtualización. La red 110 puede ser una LAN, una red de área amplia ("WAN"), Internet o cualquier otra topología de red que conecte el ordenador 102 al servidor 112 de aplicación virtual.

Los componentes de software del programa 104 de aplicación pueden ser almacenados en un paquete 114 de aplicación virtualizada situado en un dispositivo de almacenamiento accesible por el servidor 112 de aplicación virtual. De acuerdo con las realizaciones, el paquete 114 de aplicación virtualizada consiste en una serie de bloques de datos que contienen información sobre la estructura del programa de aplicación, así como los archivos de componentes individuales y otros elementos de la aplicación. El paquete 114 de aplicación virtualizada puede contener además metadatos con respecto a la ubicación y configuración de los recursos locales y remotos utilizados por el programa 104 de aplicación durante la ejecución. El administrador del programa 104 de aplicación puede crear el paquete 114 de aplicación virtualizada realizando una instalación típica de la aplicación en un servidor de administración y registrando los cambios realizados en el sistema de archivos local, el registro y otros, por ejemplo. Los bloques en el paquete 114 de aplicación virtualizada pueden luego transmitirse al entorno 106 de aplicación virtual para permitir que el programa 104 de aplicación se ejecute en el ordenador 102.

El entorno 106 de aplicación virtual puede crear un entorno de ejecución virtual independiente, que se refiere como una "caja de arena de la aplicación", para ejecutar cada programa 104 de aplicación transmitido desde el servidor 112 de aplicación virtual. La zona de pruebas de la aplicación permite que los componentes del programa 104 de aplicación se ejecuten de forma aislada del resto del sistema. El entorno 106 de aplicación virtual puede proporcionar además una capa 116 de virtualización que abstrae el acceso a los recursos 118A locales y a los recursos 118B remotos (referidos aquí en general como recursos 118) utilizados por el programa 104 de aplicación durante la ejecución. Los recursos 118 pueden incluir memoria del sistema, tiempo del procesador local o subprocesos de procesamiento, archivos almacenados en un sistema de archivos, datos almacenados en una base de datos de registro, servicios de aplicaciones, servicios de presentación, servicios de bases de datos y similares disponibles localmente en el ordenador 102 o de forma remota a través de la red 110.

El programa 104 de aplicación puede acceder a los recursos 118 locales y remotos a través de interfaces 120 de programación de aplicaciones de recursos ("API") implementadas por un sistema 122 operativo o de otras bibliotecas de software estándar instalado en el ordenador 102. De acuerdo con las realizaciones, la capa 116 de virtualización abstrae las API 120 de recursos para supervisar y controlar las solicitudes de acceso a los recursos 118 locales y remotos por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual. Además, la capa 116 de virtualización puede registrar el acceso a los recursos 118 mediante el programa 104 de

aplicación en un registro 124 de acceso a recursos. El registro 124 de acceso a recursos puede comprender un archivo de registro en el sistema de archivos local, una serie de tablas de base de datos en un servidor de base de datos remoto, una combinación de los dos, o cualquier otro sistema de almacenamiento de datos accesible por el ordenador 102.

El registro 124 de acceso a recursos puede contener un registro de eventos 126 de acceso a recursos. Los eventos 126 de acceso a recursos pueden incluir detalles de las llamadas a las API 120 de recursos realizadas por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual. Cada uno de los eventos 126 de acceso a recursos puede incluir una marca de tiempo que indique cuándo ocurrió el acceso a los recursos, un identificador de la API 120 de recursos individual llamada y un número de valores de parámetros que indican el tipo de recurso, la ubicación u otros aspectos de los recursos 118 locales o remotos siendo accedidos. Los eventos 126 de acceso a recursos pueden almacenarse como entradas en un archivo de registro, como filas en una tabla de base de datos, como objetos en un diccionario, o en cualquier otra estructura de datos o formato conocido en la técnica.

En una realización, el registro 124 de acceso a recursos contiene además los patrones 128 de acceso a recursos. Los patrones 128 de acceso a recursos pueden comprender patrones de acceso a recursos por el programa 104 de aplicación que ocurren regularmente. Por ejemplo, el programa 104 de aplicación puede leer una clave de registro particular en el momento T1 y luego escribir en un archivo particular ubicado en un sistema de archivos remoto en el tiempo T1 + 240 ms. Además, este patrón de lectura de la clave de registro y escritura en el archivo puede ocurrir más de una vez, ya sea en respuesta a un evento o condición en particular, periódicamente o en un momento determinado del día, por ejemplo. Los patrones 128 de acceso a recursos pueden establecerse entre llamadas API específicas, entre llamadas API para recursos específicos, o entre llamadas API para cantidades específicas de recursos, como una cantidad de memoria asignada o una cantidad de hilos iniciados, por ejemplo.

Los patrones 128 de acceso a recursos se pueden establecer a partir de los eventos 126 de acceso a recursos recogidos durante un periodo de tiempo usando técnicas de reconocimiento de patrones. Por ejemplo, se puede determinar que un subconjunto de tipos de eventos es importante, y se pueden utilizar técnicas de aprendizaje bayesiano para establecer los patrones 128 de acceso a recursos a través de esos tipos de eventos 126 de acceso a recursos dentro del período de recolección. Los patrones 128 de acceso a recursos generados pueden almacenarse como cadenas de Markov o árboles de probabilidad que indican una probabilidad relativa de ocurrencia entre los eventos de acceso a recursos, por ejemplo.

De acuerdo con una realización, los patrones 128 de acceso a recursos puede generar casi en tiempo real en el equipo desde el registro de eventos 126 de acceso a recursos en un proceso de fondo que se ejecuta en paralelo a la ejecución del programa 104 de aplicación. En otra realización, los eventos 126 de acceso a recursos registrados por la capa 116 de virtualización en varios ordenadores 102 que ejecutan el programa 104 de aplicación en el entorno 106 de aplicación virtual pueden agregarse en una ubicación central. Los datos de eventos agregados pueden ser genéricos, mediante la eliminación de rutas de recursos dependientes del ordenador, por ejemplo, y los patrones 128 de acceso a recursos pueden establecerse a partir de los datos de eventos agregados y genéricos. Los patrones 128 de acceso a recursos generados pueden enviarse luego a cada uno de los ordenadores 102 que ejecutan el programa 104 de aplicación para que los patrones se utilicen en la predicción de fallos de la aplicación, como se describirá con más detalle a continuación con respecto a la figura 2.

En una realización adicional, el registro 124 de acceso a recursos también contiene los datos 130 de escritura de recursos. Los datos 130 de escritura de recursos pueden contener un registro de llamadas del programa 104 de aplicación a las API 120 de recursos que agregan, modifican o eliminan datos, como una escritura de valor de registro o una escritura de memoria intermedia de E/S. Los datos 130 de escritura de recursos pueden contener una "copia profunda" de los punteros o parámetros de tipo de estructura para incluir los datos que se están escribiendo. Además, cualquier nombre de archivo, nombre de clave, dirección u otros parámetros de ubicación pueden generarse utilizando el contexto actual del programa 104 de aplicación de ejecución. Se apreciará que los datos 130 de escritura de recursos y los eventos 126 de acceso a recursos pueden integrarse en un único archivo de registro u otra estructura en el registro 124 de acceso a recursos. Los datos 130 de escritura de recursos pueden utilizarse para restaurar el estado de la aplicación del programa 104 de aplicación durante la recuperación de un fallo de la aplicación, como se describirá con más detalle a continuación con respecto a la figura 4.

Con referencia ahora a las figuras 2-4, se proporcionarán detalles adicionales con respecto a las realizaciones presentadas en la presente memoria. Debe apreciarse que las operaciones lógicas descritas con respecto a las figuras 2-4 se implementan (1) como una secuencia de actos implementados por ordenador o módulos de programa que se ejecutan en un sistema informático y/o (2) como circuitos lógicos de máquinas interconectados o módulos de circuitos dentro del sistema informático. La implementación es una cuestión de elección que depende del rendimiento y otros requisitos del sistema informático. Por consiguiente, las operaciones lógicas descritas en la presente memoria se denominan, de manera diversa, operaciones, dispositivos estructurales, actos o módulos. Estas operaciones, dispositivos estructurales, actos y módulos pueden implementarse en software, firmware, lógica digital para fines especiales y cualquier combinación de los mismos. También debe tenerse en cuenta que se pueden realizar más o menos operaciones de las que se muestran en las figuras y se describen en la presente memoria. Las operaciones también pueden realizarse en un orden diferente al descrito.

La figura 2 ilustra una rutina 200 para predecir posibles condiciones de error en el programa 104 de aplicación en base a una desviación de los patrones establecidos de acceso a los recursos, de acuerdo con las realizaciones descritas en el presente documento. La rutina 200 puede realizarse mediante una combinación de la capa 116 de virtualización que se ejecuta en el entorno 106 de aplicación virtual en el ordenador 102 y/o en otros módulos que se ejecutan en el ordenador o en servidores de aplicaciones centralizados. Se apreciará que la rutina 200 también puede ser realizada por otros módulos o componentes que se ejecutan en otros dispositivos informáticos, o por cualquier combinación de módulos, componentes y dispositivos informáticos.

La rutina 200 comienza en la operación 202, donde la capa 116 de virtualización monitoriza el acceso a recursos 118 locales y remotos por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual, y los registros de estos accesos en el registro 124 de acceso a recursos. Como se describió anteriormente con respecto a la figura 1, la capa 116 de virtualización puede registrar los detalles de las llamadas del programa 104 de aplicación a las API 120 de recursos como eventos 126 de acceso a recursos, incluida la marca de tiempo que indica cuándo ocurrió el acceso al recurso, un identificador del recurso individual API llamado y una serie de valores de parámetros que indican el tipo de recurso, la ubicación u otros aspectos de los recursos locales o remotos a los que se accede.

A partir de la operación 202, la rutina 200 prosigue a la operación 204, donde se establecen los patrones 128 de acceso a recursos. Se apreciará que, durante un cierto período de tiempo, la capa 116 de virtualización puede registrar un número significativo de eventos 126 de acceso a recursos en el registro 124 de acceso a recursos. Como se describió anteriormente con respecto a la figura 1, la capa 116 de virtualización o algún otro módulo o proceso puede utilizar los eventos 126 de acceso a recursos registrados para establecer los patrones 128 de acceso a recursos. Por ejemplo, la capa 116 de virtualización puede utilizar técnicas de reconocimiento de patrones, como las redes bayesianas, para establecer las probabilidades relativas de ocurrencia entre dos o más eventos de acceso a recursos. Los patrones 128 de acceso a recursos establecidos pueden almacenarse entonces como cadenas de Markov o árboles de probabilidad en el registro 124 de acceso a recursos.

Los patrones 128 de acceso a recursos pueden generarse casi en tiempo real en el ordenador 102 por la capa 116 de virtualización. Alternativamente, los eventos 126 de acceso a los recursos registrados pueden ser agregados desde una serie de ordenadores 102 en una ubicación central, generados y utilizados para establecer patrones 128 de acceso a recursos en múltiples instancias del programa 104 de aplicación que se ejecutan en el entorno 106 de aplicación virtual en los ordenadores. Los patrones 128 de acceso a recursos genéricos establecidos a partir de los eventos 126 de acceso a recursos agregados pueden utilizarse para predecir las condiciones de error en cualquier ordenador 102 que ejecute el programa 104 de aplicación de la manera descrita en la presente memoria.

La rutina 200 prosigue desde la operación 204 a la operación 206, donde la capa 116 de virtualización detecta una desviación de los patrones 128 de acceso de recursos establecidos por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual. Por ejemplo, la capa 116 de virtualización puede detectar una secuencia de llamadas a la API de recursos que tiene una probabilidad de ocurrencia por debajo de un umbral específico basado en el análisis bayesiano de los patrones 128 de acceso a recursos. De manera similar, la capa 116 de virtualización puede detectar una secuencia de llamadas a la API de recursos que tiene una alta probabilidad de ocurrencia con una condición de error conocida, según lo establecido en los patrones 128 de acceso a recursos. En una realización, si la probabilidad de la secuencia de llamadas a la API de recursos no cae por debajo del umbral específico, entonces la capa 116 de virtualización registra los eventos 126 de acceso a recursos correspondientes para que los patrones 128 de acceso a recursos puedan actualizarse con nuevas probabilidades en el proceso de origen descrito anteriormente. De esta manera, los patrones 128 de acceso a recursos pueden actualizarse continuamente durante la ejecución del programa 104 de aplicación en el entorno 106 de aplicación virtual.

Si se detecta una desviación de los patrones 128 de acceso de recursos establecidos por el programa 104 de aplicación, la rutina 200 prosigue desde la operación 206 a la operación 208, donde la capa 116 de virtualización plantea una alerta con respecto a la desviación en el patrón. La alerta puede dirigirse a un usuario o administrador del programa 104 de aplicación. La alerta puede enviarse por correo electrónico, mensaje de texto o cola de mensajes del sistema; planteado como un evento de nivel de sistema; registrado en una aplicación o registro de eventos del sistema, o transmitido de otra manera al administrador a través de un sistema de mensajería accesible por el ordenador 102. Esta alerta proactiva puede dar al administrador la oportunidad de tomar acciones de diagnóstico o correctivas rápidamente, reduciendo o quizás evitando el tiempo de inactividad y la pérdida de datos a raíz de una posible condición de error pendiente. A partir de la operación 208, la rutina 200 termina.

La figura 3 ilustra una rutina 300 para correlacionar los eventos 126 de acceso a recursos con una condición de error conocida en el programa 104 de aplicación para permitir el diagnóstico del error, de acuerdo con las realizaciones descritas en la presente memoria. La rutina 300 puede realizarse mediante una combinación de la capa 116 de virtualización que se ejecuta en el entorno 106 de aplicación virtual en el ordenador 102 y/o en otros módulos que se ejecutan en el ordenador o en servidores de aplicaciones centralizados. Se apreciará que la rutina 300 también puede ser realizada por otros módulos o componentes que se ejecutan en otros dispositivos informáticos, o por cualquier combinación de módulos, componentes y dispositivos informáticos.

La rutina 300 comienza en la operación 302, donde la capa 116 de virtualización monitoriza el acceso a recursos 118 locales y remotos por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual y los

registros de estos accesos al registro 124 de acceso a recursos, en la forma descrito anteriormente con respecto a la operación 202. La rutina 300 luego pasa a la operación 304, donde se detecta una condición de error en el programa 104 de aplicación. Por ejemplo, la condición de error puede ser detectada en el programa 104 de aplicación por un usuario o administrador de la aplicación a través de medios tradicionales, como un cuadro de diálogo emergente de error, un evento registrado en una aplicación o registro de eventos del sistema, o similar.

A partir de la operación 304, la rutina 300 prosigue a la operación 306, donde los eventos 126 de acceso a recursos en el registro 124 de acceso a recursos se correlacionan con la condición de error detectado. El administrador puede proporcionar un tiempo de ocurrencia de la condición de error, o el tiempo de ocurrencia de la condición de error puede identificarse a partir de llamadas específicas a las API 120 de recursos registradas en los eventos 126 de acceso a recursos. La capa 116 de virtualización u otro módulo puede entonces identificar un subconjunto de eventos 126 de acceso a recursos en el registro 124 de acceso a recursos en una proximidad temporal al momento en que ocurre la condición de error. Por ejemplo, todos los eventos 126 de acceso a recursos que ocurrieron dentro de una ventana de 10 segundos antes de la condición de error pueden estar correlacionados con la condición de error.

En una realización, solo las secuencias de llamadas a la API de recursos registrados en los eventos 126 de acceso a recursos dentro de la proximidad temporal de la condición de error que se desvían de los patrones 128 de acceso de recursos establecidos están correlacionados con la condición de error. Por ejemplo, la capa 116 de virtualización u otro módulo puede identificar secuencias de llamadas a la API de recursos registradas en los eventos 126 de acceso a recursos dentro de los 10 segundos posteriores a la aparición de la condición de error que tienen una probabilidad de ocurrencia por debajo de un umbral específico basado en el análisis bayesiano de los patrones 128 de acceso a recursos en el registro 124 de acceso a recursos. Se apreciará que el umbral de probabilidad especificado para correlacionar desviaciones en los patrones 128 de acceso a recursos con una condición de error puede ser mayor que el umbral de probabilidad para predecir una condición de error en base a las desviaciones en los patrones de acceso a recursos descritos anteriormente con respecto a la operación 206.

La rutina 300 procede entonces de la operación 306 a la operación 308, donde los eventos 126 de acceso a recursos correlacionados con la condición de error se muestran al usuario o administrador del programa 104 de aplicación. Los eventos 126 de acceso a recursos correlacionados se pueden mostrar a través de un diálogo de interfaz de usuario o en un informe transmitido por correo electrónico, mensaje de texto, cola de mensajes del sistema o similares. Proporcionar eventos 126 de acceso a recursos desde el registro 124 de acceso a recursos que se correlacionan en el tiempo con una condición de error particular puede permitir al usuario o administrador diagnosticar rápidamente la causa de la condición de error e implementar las acciones de recuperación apropiadas para reducir el tiempo improductivo y la pérdida de datos. A partir de la operación 308, la rutina 300 termina.

La figura 4 ilustra una rutina 400 para recuperarse de una condición de error en un programa 104 de aplicación, de acuerdo con las realizaciones descritas en la presente memoria. La rutina 400 puede realizarse mediante una combinación de la capa 116 de virtualización que se ejecuta en el entorno 106 de aplicación virtual en el ordenador 102 y/o en otros módulos que se ejecutan en el ordenador o en servidores de aplicaciones centralizados. Se apreciará que la rutina 400 también puede ser realizada por otros módulos o componentes que se ejecutan en otros dispositivos informáticos, o por cualquier combinación de módulos, componentes y dispositivos informáticos.

La rutina 400 comienza en la operación 402, donde los monitores capa 116 de virtualización acceso a recursos 118 locales y remotos por el programa 104 de aplicación que se ejecuta en el entorno 106 de aplicación virtual y los registros de estos accesos al registro 124 de acceso a recursos, en la forma descrito anteriormente con respecto a la operación 202. Además, la capa 116 de virtualización registra los datos 130 de escritura de recursos para las llamadas del programa 104 de aplicación a las API 120 de recursos que agregan, modifican o eliminan datos, como se describió anteriormente con respecto a la figura 1. Los datos 130 de escritura de recursos pueden incluir una copia profunda de los punteros o los parámetros de tipo de estructura especificados en las llamadas a la API, y los datos pueden procesarse aún más utilizando el contexto actual del programa 104 de aplicación de ejecución para generar el nombre de archivo, el nombre de la clave, la dirección u otros parámetros de ubicación en los datos 130 de escritura de recursos.

A partir de la operación 402, la rutina 400 prosigue hacia la operación 404, donde el programa 104 de aplicación fallo debido a una condición de error. Por ejemplo, el programa 104 de aplicación puede fallar debido a un error de software, un error de usuario, datos de entrada incorrectos, recursos no disponibles, un fallo de hardware en el ordenador 102 o similar. Al fallar el programa 104 de aplicación, la rutina 400 pasa a la operación 406, donde el programa 104 de aplicación se reinicia. El programa 104 de aplicación puede ser reiniciado automáticamente por la capa 116 de virtualización u otro módulo que se ejecute en el ordenador 102, o puede ser reiniciado manualmente por un administrador del sistema en el mismo ordenador u otro sistema de ordenador con una configuración similar.

La rutina 400 procede de la operación 406 a la operación 408, donde la capa 116 de virtualización de ejecución en el entorno 106 de aplicación virtual en el equipo 102 en el que el programa 104 de aplicación se reinició repeticiones eventos de acceso determinado recurso 126 registrados en el registro 124 de acceso a recursos para restaurar el estado de la aplicación a un punto antes de que se produjera el fallo. Por ejemplo, la capa 116 de virtualización puede reproducir todos los eventos 126 de acceso a recursos correspondientes a las llamadas a las API 120 de

recursos que escriben en ubicaciones de almacenamiento volátiles o almacenadas en caché, como direcciones de memoria del sistema, memorias intermedias de E/S, archivos almacenados en caché o similares.

En una realización adicional, la capa 116 de virtualización puede reproducir todos los eventos 126 de acceso a recursos que corresponden a la escritura de datos que se produjo desde una última instantánea del estado de la aplicación, o "punto de control", fue tomada y almacenado por la capa de virtualización y/o programa 104 de aplicación antes del fallo de la aplicación. La capa 116 de virtualización puede utilizar los datos 130 de escritura de recursos en el registro 124 de acceso a recursos para reproducir los eventos de acceso a recursos seleccionados 126, a fin de garantizar que se escriban los datos adecuados para restaurar el estado de la aplicación. El registro de eventos 126 de acceso a recursos que agregan, modifican o eliminan datos junto con los datos de escritura de recursos correspondientes 130 para reproducir las escrituras para restaurar el estado de la aplicación puede permitir una recuperación más rápida de los fallos de la aplicación, lo que reduce el tiempo de inactividad de la aplicación. A partir de la operación 408, la rutina 400 termina.

Aunque la presente descripción se describe en el contexto de un entorno 106 de aplicación virtualizada, se apreciará que los procedimientos presentados en el presente documento para diferenciar el funcionamiento normal de un programa de aplicación de condiciones de error para predecir, diagnosticar y recuperarse de los fallos de aplicación puede implementarse en cualquier otro entorno de aplicación donde se pueda monitorear el acceso de los programas de aplicación 104 a los recursos 118 locales y remotos. Por ejemplo, se puede implementar un módulo similar al de la capa 116 de virtualización que utiliza procedimientos conocidos en la técnica para enlazar los API 120 de recursos implementados por el sistema operativo 122 para monitorear las solicitudes de recursos 118 locales y remotos mediante un programa de aplicaciones que se ejecuta localmente 104 fuera de un entorno 106 de aplicación virtual. El módulo puede registrar los accesos a los recursos y establecer patrones comunes de uso de recursos por parte del programa 104 de aplicación. Una vez que se establecen dichos patrones de acceso a los recursos, el módulo puede continuar supervisando el uso de los recursos por el programa 104 de aplicación y proporcionar una advertencia o alerta cuando los patrones cambian, de la manera descrita en la presente memoria.

La figura 5 muestra una arquitectura de ordenador de ejemplo para un ordenador 500 capaz de ejecutar los componentes de software descritos aquí para diferenciar el funcionamiento normal de un programa de aplicación de las condiciones de error para predecir, diagnosticar y recuperarse de fallos de la aplicación, de la manera presentada anteriormente. La arquitectura del ordenador que se muestra en la figura 5 ilustra un ordenador de servidor convencional, ordenador de escritorio, ordenador portátil, notebook, PDA, teléfono inalámbrico u otro dispositivo de computación, y puede utilizarse para ejecutar cualquier aspecto de los componentes de software aquí presentados que se ejecutan en el ordenador 102 u otro dispositivo informático.

La arquitectura de ordenador se muestra en la figura 5 incluye una o más unidades 502 centrales de proceso ("CPU"). Las CPU 502 pueden ser procesadores estándar que realizan las operaciones aritméticas y lógicas necesarias para el funcionamiento del ordenador 500. Las CPU 502 realizan las operaciones necesarias mediante la transición de un estado físico discreto al siguiente a través de la manipulación de elementos de conmutación que diferencian y cambian estos estados. Los elementos de conmutación generalmente pueden incluir circuitos electrónicos que mantienen uno de dos estados binarios, tales como biestables, y circuitos electrónicos que proporcionan un estado de salida basado en la combinación lógica de los estados de uno o más elementos de conmutación, como las puertas lógicas. Estos elementos básicos de conmutación pueden combinarse para crear circuitos lógicos más complejos, incluidos registros, sumadores-restadores, unidades lógicas aritméticas, unidades de punto flotante y otros elementos lógicos.

La arquitectura de ordenador incluye además una memoria 508 de sistema, que incluye una memoria 514 de acceso aleatorio ("RAM") y una memoria 516 de solo lectura ("ROM"), y un sistema 504 de bus que acopla la memoria a la CPU 502. Un sistema básico de entrada/salida que contiene las rutinas básicas que ayudan a transferir información entre elementos dentro del ordenador 500, como durante el inicio, se almacena en la ROM 516. El ordenador 500 también incluye un dispositivo 510 de almacenamiento masivo para almacenar un sistema 122 operativo, programas de aplicación y otros módulos de programa, que se describen con mayor detalle en la presente memoria.

El dispositivo 510 de almacenamiento masivo está conectado a la CPU 502 a través de un controlador de almacenamiento masivo (no mostrado) conectado al bus 504. El dispositivo 510 de almacenamiento masivo proporciona almacenamiento no volátil para el ordenador 500. El ordenador 500 puede almacenar información en el dispositivo 510 de almacenamiento masivo al transformar el estado físico del dispositivo para reflejar la información que se almacena. La transformación específica del estado físico puede depender de varios factores, en diferentes implementaciones de esta descripción. Los ejemplos de tales factores pueden incluir, pero no se limitan a, la tecnología utilizada para implementar el dispositivo de almacenamiento masivo ya sea que el dispositivo de almacenamiento masivo se caracterice como almacenamiento primario o secundario, y similares.

Por ejemplo, el ordenador 500 puede almacenar información en el dispositivo 510 de almacenamiento masivo emitiendo instrucciones al controlador de almacenamiento masivo para alterar las características magnéticas de una ubicación particular dentro de una unidad de disco magnético, las características de reflexión o de refracción de una ubicación particular en un dispositivo de almacenamiento óptico, o las características eléctricas de un condensador, transistor u otro componente discreto en particular en un dispositivo de almacenamiento de estado sólido. Otras

transformaciones de los medios físicos son posibles sin apartarse del alcance y el espíritu de la presente descripción. El ordenador 500 puede leer más información del dispositivo 510 de almacenamiento masivo al detectar los estados físicos o las características de una o más ubicaciones particulares dentro del dispositivo de almacenamiento masivo.

Como se ha mencionado brevemente más arriba, un número de módulos de programa y los archivos de datos se pueden almacenar en el dispositivo 510 de almacenamiento masivo y la RAM 514 del ordenador 500, incluyendo un sistema 518 operativo adecuado para controlar el funcionamiento de un ordenador. El dispositivo 510 de almacenamiento masivo y la RAM 514 también pueden almacenar uno o más módulos de programa. En particular, el dispositivo 510 de almacenamiento masivo y la RAM 514 pueden almacenar la capa 116 de virtualización, que se describió en detalle anteriormente con respecto a la figura 1. El dispositivo 510 de almacenamiento masivo y la RAM 514 también pueden almacenar otros tipos de módulos de programa o datos.

Además del dispositivo 510 de almacenamiento masivo se ha descrito anteriormente, el ordenador 500 puede tener acceso a otros medios legibles por ordenador para almacenar y recuperar información, tal como módulos de programa, estructuras de datos, u otros datos. Los expertos en la técnica deberían apreciar que los medios legibles por ordenador pueden ser cualquier medio disponible al que pueda acceder el ordenador 500, incluidos los medios de almacenamiento legibles por ordenador y los medios de comunicación. Los medios de comunicación incluyen señales transitorias. Los medios legibles por ordenador incluyen medios volátiles y no volátiles, extraíbles y no extraíbles implementados en cualquier procedimiento o tecnología para el almacenamiento de información tal como instrucciones legibles por ordenador, estructuras de datos, módulos de programas u otros datos. Por ejemplo, los medios de almacenamiento legibles por ordenador incluyen, pero no se limitan a, RAM, ROM, EEPROM, memoria flash u otra tecnología de memoria, CDROM, discos versátiles digitales (DVD), HD-DVD, BLU-RAY u otro almacenamiento de disco óptico, casetes magnéticos, cinta magnética, almacenamiento en disco magnético u otros dispositivos de almacenamiento magnético que pueden almacenar los datos deseados y a los que se puede acceder desde el ordenador 500.

El medio de almacenamiento legible por ordenador puede ser codificado con instrucciones ejecutables por ordenador que, cuando se carga en el ordenador 500, puede transformar el sistema informático de un sistema informático de propósito general en un ordenador de propósito especial capaz de implementar las realizaciones descritas en el presente documento. Las instrucciones ejecutables por ordenador pueden codificarse en el medio de almacenamiento legible por ordenador al alterar las características eléctricas, ópticas, magnéticas u otras características físicas de ubicaciones particulares dentro de los medios. Estas instrucciones ejecutables por ordenador transforman el ordenador 500 al especificar cómo las CPU 502 hacen la transición entre los estados, como se describió anteriormente. De acuerdo con una realización, el ordenador 500 puede tener acceso a medios de almacenamiento legibles por ordenador que almacenan instrucciones ejecutables por ordenador que, cuando son ejecutadas por el ordenador, realizan las rutinas 200, 300 y/o 400 para diferenciar el funcionamiento normal de un programa de aplicación de condiciones de error para predecir, diagnosticar y recuperarse de fallos en la aplicación, descritas anteriormente con respecto a las figuras 2-4.

De acuerdo con diversas formas de realización, el ordenador 500 puede operar en un entorno de red usando conexiones lógicas a dispositivos informáticos y sistemas informáticos a distancia a través de la red 110, tal como una LAN, una WAN, Internet, o una red de cualquier topología conocida en la técnica. El ordenador 500 puede conectarse a la red 110 a través de una unidad 506 de interfaz de red conectada al bus 504. Debe apreciarse que la unidad 506 de interfaz de red también puede utilizarse para conectarse a otros tipos de redes y sistemas informáticos remotos.

El ordenador 500 también puede incluir un controlador 512 de entrada/salida para recibir y de entrada de tratamiento de un número de dispositivos de entrada, que incluye un teclado, un ratón, una pantalla táctil, una pantalla táctil, un lápiz electrónico, u otro tipo de entrada dispositivo. De manera similar, el controlador de entrada/salida 512 puede proporcionar salida a un dispositivo de visualización, tal como un monitor de ordenador, una pantalla de panel plano, un proyector digital, una impresora, un plóter u otro tipo de dispositivo de salida. Se apreciará que el ordenador 500 puede no incluir todos los componentes mostrados en la figura 5, puede incluir otros componentes que no se muestran explícitamente en la figura 5, o puede utilizar una arquitectura completamente diferente a la que se muestra en la figura 5.

En base a lo anterior, debe apreciarse que las tecnologías para diferenciar el funcionamiento normal de un programa de aplicación de las condiciones de error para predecir, diagnosticar y recuperarse de los fallos de la aplicación se proporcionan en el presente documento. Aunque el tema presentado aquí se ha descrito en un lenguaje específico para características estructurales de ordenador, actos metodológicos y medios de almacenamiento legibles por ordenador, debe entenderse que la invención definida en las reivindicaciones adjuntas no está necesariamente limitada a las características, actos, o los medios específicos descritos en la presente memoria. Más bien, las características, actos y medios específicos se divulgan como formas de ejemplo de implementación de las reclamaciones.

El objeto descrito anteriormente se proporciona solamente a modo de ilustración y no debe interpretarse como limitante. Se pueden realizar varias modificaciones y cambios al objeto descrito en la presente memoria sin seguir

las realizaciones de ejemplo y las aplicaciones ilustradas y descritas, y sin apartarse del alcance de la presente invención, que se expone en las siguientes reivindicaciones.

REIVINDICACIONES

1. Un procedimiento (200) implementado por ordenador, para predecir una posible condición de error en un programa (104) de aplicación que se ejecuta en un entorno (106) de aplicación virtual, en el que el programa (104) de aplicación accede a recursos (118) locales y remotos a través de interfaces (120) de programación de aplicaciones de recursos , comprendiendo el procedimiento ejecutar instrucciones en uno o más ordenadores para realizar las operaciones de:

registrar (202) eventos (126) de acceso a recursos iniciados por el programa (104) de aplicación que se ejecuta en el entorno (106) de aplicación virtual, en el que cada uno de los eventos (126) de acceso a recursos incluye una marca de tiempo que indica cuándo ocurrió el acceso al recurso;
establecer (204) patrones de acceso a recursos (128) a partir de los eventos (126) de acceso a recursos registrados, en el que los patrones (128) de acceso a recursos se establecen a partir de los eventos (126) de acceso a recursos recopilados durante un período de tiempo;
detectar (206) una desviación de los patrones (128) de acceso a recursos establecidos por el programa (104) de aplicación; y
alertar (208) a un usuario o administrador del programa (104) de aplicación de una posible condición de error en base a la desviación detectada.

2. El procedimiento (200) implementado por ordenador de la reivindicación 1, que comprende además el registro de datos de valores de parámetros especificados en llamadas a las API de recursos para agregar, modificar o eliminar datos mediante el programa (104) de aplicación.

3. El procedimiento (200) implementado por ordenador de la reivindicación 2, que comprende además utilizar los datos registrados de los valores de parámetros especificados en las llamadas a las API de recursos para reproducir los eventos (126) de acceso a los recursos correspondientes para agregar, modificar o eliminar datos con el fin de restaurar un estado de la aplicación del programa (104) de aplicación después de un fallo de la aplicación.

4. El procedimiento (200) implementado por ordenador de la reivindicación 1, en el que los patrones (128) de acceso a recursos se establecen a partir de los eventos (126) de acceso a recursos registrados utilizando técnicas de aprendizaje bayesiano.

5. El procedimiento (200) implementado por ordenador de la reivindicación 4, en el que los patrones (128) de acceso a recursos se almacenan como cadenas de Markov o árboles de probabilidad que indican una probabilidad relativa de ocurrencia entre los eventos (126) de acceso a recursos.

6. El procedimiento (200) implementado por ordenador de la reivindicación 4, en el que la detección de la desviación de los patrones (128) de acceso a recursos establecidos comprende la detección de una secuencia de eventos (126) de acceso a recursos que tienen una probabilidad de ocurrencia inferior a una probabilidad de umbral especificada en base al análisis bayesiano de los patrones (128) de acceso a recursos.

7. El procedimiento (200) implementado por ordenador de la reivindicación 4, en el que las secuencias de eventos (126) de acceso a recursos que tienen una probabilidad de ocurrencia inferior a una probabilidad umbral en base al análisis bayesiano de los patrones (128) de acceso a recursos se correlacionan con una condición de error detectada del programa (104) de aplicación en base a una proximidad temporal a un tiempo de ocurrencia de la condición de error.

8. Un medio de almacenamiento legible por ordenador codificado con instrucciones ejecutables por ordenador que, cuando se ejecutan por un ordenador, hace que el ordenador realice el procedimiento de una de las reivindicaciones 1 a 7.

9. El medio de almacenamiento legible por ordenador de la reivindicación 8, codificado con instrucciones adicionales ejecutables por ordenador que hacen que el ordenador:

detecte (304) una condición de error en el programa (104) de aplicación;
determine un tiempo de ocurrencia de la condición de error;
correlacione (306) uno o más de los eventos (126) de acceso a recursos registrados en el registro (124) de acceso a recursos en débase a la proximidad temporal al momento de ocurrencia de la condición de error; y
muestre (308) el uno o más eventos (126) de acceso a recursos correlacionados a un usuario o administrador del programa (104) de aplicación.

10. El medio de almacenamiento legible por ordenador de la reivindicación 8, en el que el uno o más eventos (126) de acceso a recursos correlacionados comprenden secuencias de eventos de acceso a recursos que tienen una probabilidad de ocurrencia menor que una probabilidad de umbral en base al análisis bayesiano de los patrones (128) de acceso a recursos establecidos.

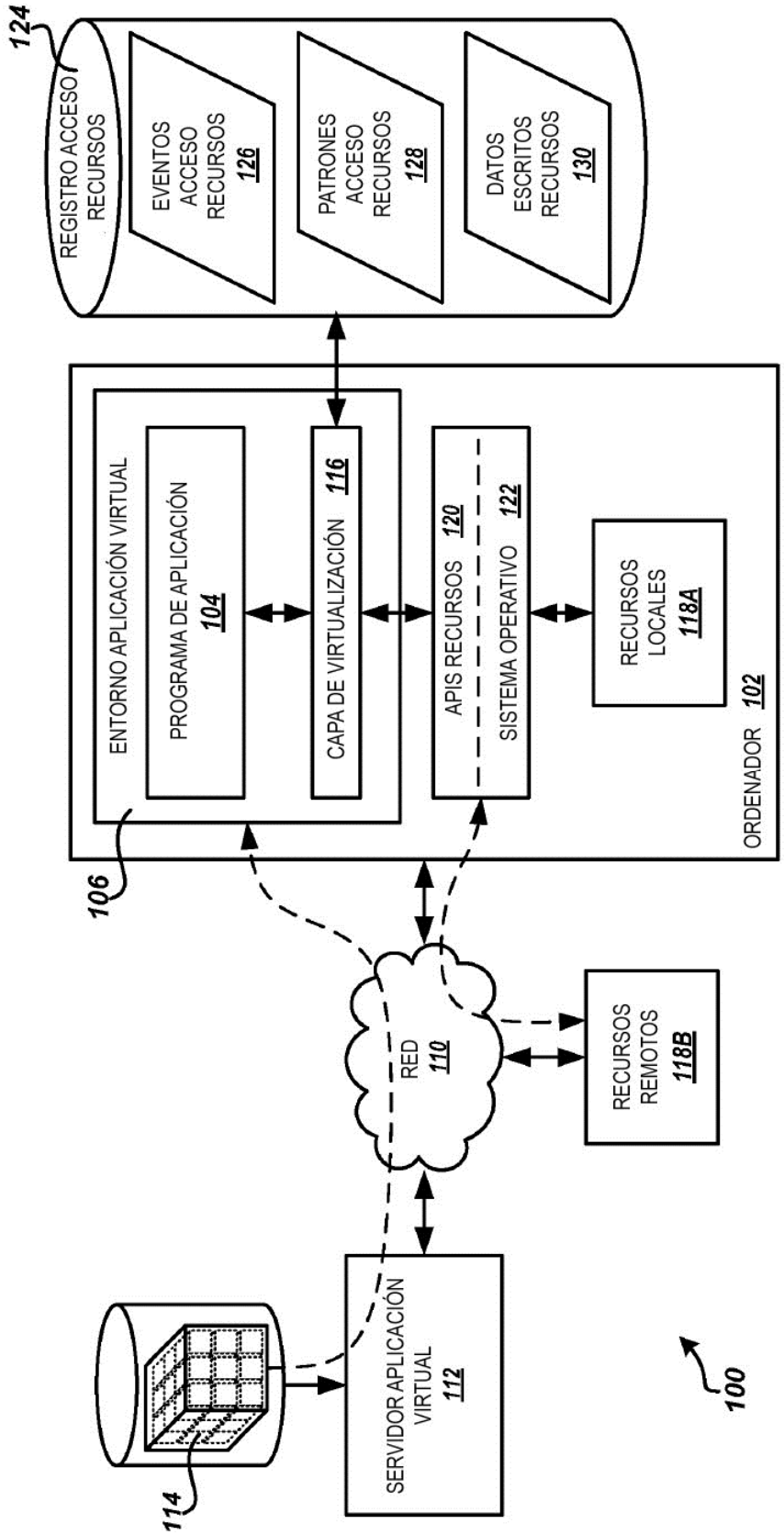
124
REGISTRO ACCESO RECURSOS
EVENTOS ACCESO RECURSOS 126
PATRONES ACCESO RECURSOS 128
DATOS ESCRITOS RECURSOS 130
ENTORNO APLICACIÓN VIRTUAL
PROGRAMA DE APLICACIÓN 104
CAPA DE VIRTUALIZACIÓN 116
APIS RECURSOS 120
SISTEMA OPERATIVO 122
RECURSOS LOCALES 118A
ORDENADOR 102
106
RED 110
RECURSOS REMOTOS 118B
SERVIDOR APLICACIÓN VIRTUAL 112
114
100

FIG. 1

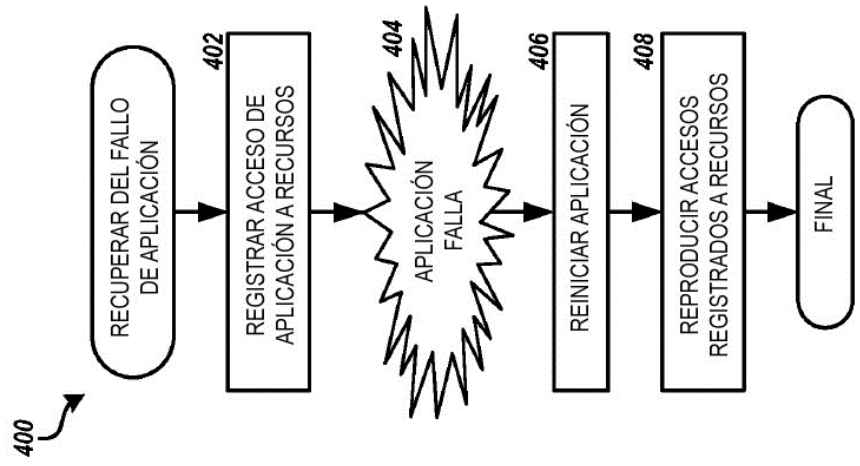

FIG. 4

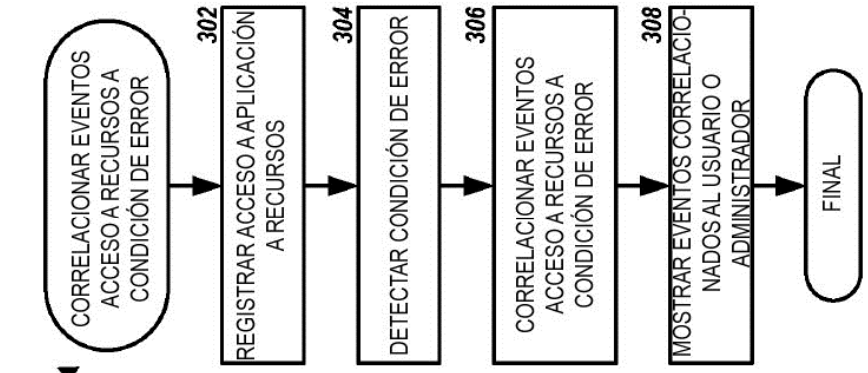

FIG. 3

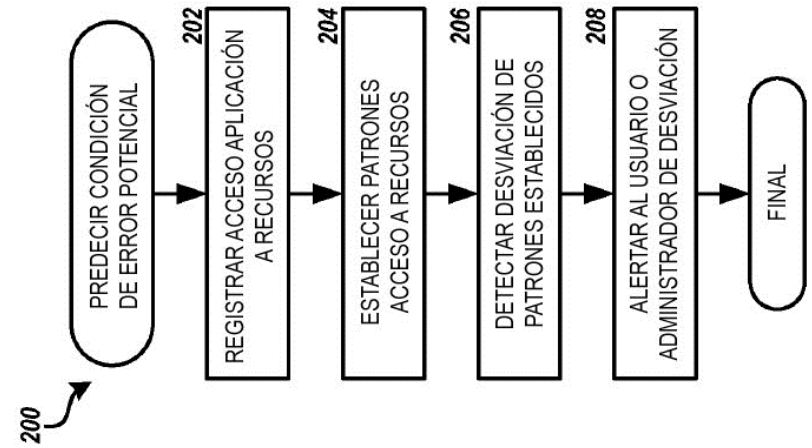

FIG. 2

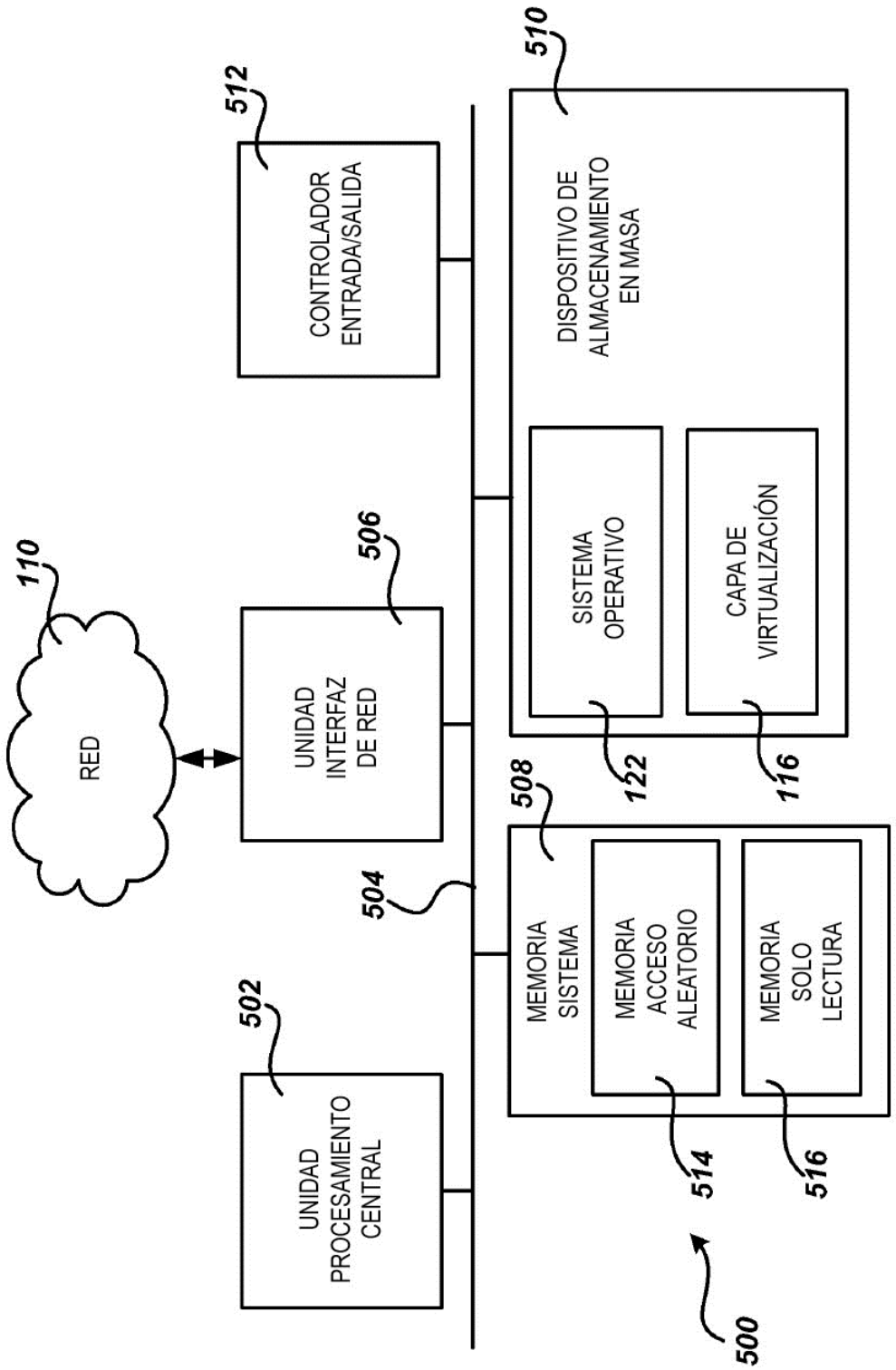
110
RED
506
UNIDAD INTERFAZ DE RED
512
CONTROLADOR ENTRADA/SALIDA
510
DISPOSITIVO DE ALMACENAMIENTO EN MASA
SISTEMA OPERATIVO
CAPA DE VIRTUALIZACIÓN
122
116
508
MEMORIA SISTEMA
MEMORIA ACCESO ALEATORIO
MEMORIA SOLO LECTURA
514
516
504
502
UNIDAD PROCESAMIENTO CENTRAL
500

FIG. 5