

GUÍA DE TOMA DE DECISIONES MEDIANTE APRENDIZAJE AUTOMÁTICO PARA ALERTAS ORIGINARIAS DE SISTEMAS DE MONITORIZACIÓN

Antecedentes de la invención

Las pruebas y la monitorización de integración automatizada son una herramienta importante para reducir la cantidad de errores en varios sistemas de *software*. Sin embargo, a medida que estos sistemas de *software* evolucionan en complejidad y se distribuyen a través de varias plataformas diferentes, la interoperabilidad entre componentes estará sujeta a fallas intermitentes. En algunos casos, las fallas pueden ser problemas relacionados con el *software*. Por ejemplo, a medida que se implementa nuevo *software* en el sistema, un desarrollador puede introducir inadvertidamente uno o más errores en el sistema. En tales casos, el desarrollador puede necesitar identificar el error y proporcionar una o más actualizaciones al *software*. En otros casos, las fallas pueden estar relacionadas con problemas de red y/o *hardware*. Cuando se producen estos errores, pueden aparecer como errores, pero en realidad son falsos positivos. Debido a que estas fallas no son provocadas por el código recientemente implementado, el desarrollador puede invertir un tiempo innecesario investigando estas fallas.

Los ejemplos se describen con respecto a estas y otras consideraciones generales. Además, aunque se aborden problemas relativamente específicos, debe entenderse que los ejemplos no deben limitarse a resolver los problemas específicos identificados en los antecedentes de la invención.

Breve descripción de la invención

Esta divulgación generalmente se refiere a un sistema y método para determinar si un problema o falla detectada en un sistema de *software* es el resultado de un error inducido por la programación o debido a alguna otra causa. Cuando se detecta una falla u otro problema con el sistema de *software*, el sistema descrito en el presente documento guía los procesos de toma de decisiones de los desarrolladores y los ayuda a reconocer y abordar problemas de *software* (por ejemplo, errores causados por la adición de código fuente al sistema de *software*) en lugar de rastrear e intentar reparar falsos positivos. Como tal, el sistema descrito en el presente documento permite a los desarrolladores invertir su tiempo de manera más eficiente. A medida que se resuelven los problemas, los datos correspondientes a los diversos problemas detectados (por ejemplo, la causa del problema, cómo se solucionó el problema, etcétera) se retroalimentan en el sistema para proporcionar una mejor guía y modelos más precisos.

En consecuencia, en el presente documento se describe un método para determinar automáticamente si un problema detectado en un sistema de cómputo es un error. Este método incluye recibir información sobre un problema detectado y almacenar la información sobre el problema detectado en un dispositivo de almacenamiento. Se genera y analiza un vector de características para el problema detectado con el fin de determinar la probabilidad de que el problema detectado sea un error. En algunos ejemplos, el análisis se basa, al menos en parte, en el vector de características. Cuando la probabilidad está por encima de un umbral, el problema detectado se informa a un dispositivo cliente.

También se describe un sistema que comprende al menos una unidad de procesamiento y una

memoria que almacena instrucciones ejecutables por computadora que, cuando se ejecutan por al menos una unidad de procesamiento, provocan que el sistema realice un método para determinar si un problema detectado en un sistema de *software* es un error o un falso positivo. Este método incluye recibir información sobre un problema detectado y comparar una propiedad distintiva del problema detectado con una o más propiedades distintivas de los problemas detectados previamente para determinar si el problema detectado es nuevo. Cuando se determina que el problema detectado es nuevo, la información sobre el problema detectado se almacena en un dispositivo de almacenamiento. La información sobre el problema detectado también se proporciona a un componente de generación de características que genera un vector de características para el problema detectado. El vector de características puede proporcionarse a un componente de evaluación que analiza el vector de características para determinar la probabilidad de que el problema detectado sea un error. Entonces puede proporcionarse una lista ordenada de problemas detectados a un dispositivo cliente.

La presente divulgación también describe un medio de almacenamiento legible por computadora que almacena instrucciones ejecutables por computadora que, cuando se ejecutan por una unidad de procesamiento, provocan que la unidad de procesamiento realice un método que comprende generar un vector de características para un problema detectado utilizando la información asociada con el problema detectado y proporcionar el vector de características para uno o más modelos de un componente de evaluación. Uno o más modelos del componente de evaluación pueden analizar el vector de características para determinar la probabilidad de que el problema detectado sea un error. Entonces se informa la aparición del problema detectado. Una vez que se informa el problema detectado, puede recibirse una solución al problema detectado. Cuando se recibe, al menos uno de uno o más modelos se actualiza utilizando la solución al problema detectado.

Esta breve descripción de la invención se proporciona para presentar una selección de conceptos en una forma simplificada que se describe posteriormente en la descripción detallada de la invención. Esta breve descripción de la invención no está destinada a identificar características importantes o características esenciales de la materia objeto reclamada, ni está destinado a utilizarse para limitar el alcance de la materia objeto reclamada.

Breve descripción de los dibujos

Los ejemplos no limitativos y no exhaustivos se describen con referencia a las siguientes figuras.

La figura 1 ilustra un sistema para evaluar y clasificar las alertas que surgen en un sistema de *software* de acuerdo con un ejemplo.

La figura 2 ilustra un método para evaluar y clasificar los problemas que surgen en un sistema de *software* de acuerdo con un ejemplo.

La figura 3 ilustra un método para resolver automáticamente problemas que surgen en un sistema de *software* de acuerdo con los ejemplos.

La figura 4 ilustra un método para recibir retroalimentación para un problema detectado de acuerdo con los ejemplos.

La figura 5 es un diagrama de bloques que ilustra los componentes físicos ejemplares de un dispositivo de cómputo con los cuales pueden ponerse en práctica los aspectos de la divulgación.

Las figuras 6A y 6B son diagramas de bloques simplificados de un dispositivo de cómputo móvil con el cual pueden ponerse en práctica los aspectos de la presente divulgación.

La figura 7 es un diagrama de bloques simplificado de un sistema de cómputo distribuido en el cual pueden ponerse en práctica los aspectos de la presente divulgación.

La figura 8 ilustra un dispositivo de cómputo tipo tableta para ejecutar uno o más aspectos de la presente divulgación.

Descripción detallada de la invención

En la siguiente descripción detallada de la invención, se hace referencia a los dibujos adjuntos que forman parte de la misma, y en los cuales las modalidades o ejemplos específicos se muestran a modo de ilustraciones. Estos aspectos pueden combinarse, pueden utilizarse otros aspectos y pueden realizarse cambios estructurales sin apartarse de la presente divulgación. Pueden ponerse en práctica los ejemplos como métodos, sistemas o dispositivos. En consecuencia, los ejemplos pueden adoptar la forma de una implementación de *hardware*, una implementación completamente de *software* o una implementación que combina aspectos de *software* y *hardware*. Por lo tanto, la siguiente descripción detallada de la invención no debe considerarse en un sentido limitante, y el alcance de la presente divulgación se define por las reivindicaciones adjuntas y sus equivalentes.

Las pruebas y la monitorización automáticos de los sistemas de *software* son una parte integral del proceso de desarrollo de *software*. Esta divulgación describe un sistema y método que determina si un problema o falla detectada en el sistema de *software* es un error o si es un falso positivo.

Por ejemplo, en cualquier momento, muchos componentes de *software* diferentes pueden estar sujetos a diversos procesos de prueba para determinar si un cambio en un componente de *software* afecta negativamente a otro componente y/o si el cambio afecta negativamente al sistema en su conjunto. En algunos casos, la integración de varios componentes de *software* puede probarse a medida que se crea un nuevo *software* y/o cuando se modifican los componentes de *software* existentes.

Sin embargo, debido a la complejidad de un laboratorio de automatización que prueba automáticamente la integración de los componentes, pueden surgir problemas falsos e intermitentes. En algunos casos, estos problemas interrumpen la integración entre algunos componentes y dan como resultado la notificación de un falso positivo. El falso positivo puede ser comunicado al desarrollador como un error o alerta. Cuando se informan estos problemas, el desarrollador puede invertir un tiempo valioso tratando de corregir un error que se produjo sin culpa propia.

El sistema descrito en el presente documento está configurado para escanear cada error o alerta informada que se origina en el laboratorio de automatización y proporcionar a los desarrolladores asistencia para la toma de decisiones sobre los errores y/o alertas que el desarrollador debe priorizar. El sistema también puede resolver automáticamente los problemas que se presumen o que se determina que son falsos positivos.

Por ejemplo, cuando se detecta un problema, falla u otra alerta (colectivamente referidos en el presente documento como un "problema"), la información sobre el problema se transmite desde un laboratorio de automatización a una capa de almacenamiento. Un componente de generación de características accede al componente de almacenamiento y genera un vector de características para el problema utilizando la información almacenada. Entonces, el vector de características se transmite a un componente de evaluación que aloja varios modelos que se utilizan para pronosticar o determinar si un problema en particular es un falso positivo o si, en realidad, es un error.

Tal como se utiliza en el presente documento, el término "error" significa un error de *software*. Por ejemplo, el error de *software* puede ser un error en un programa o sistema de cómputo, provocado por la alteración o la adición del código fuente, que hace que el programa o sistema de cómputo produzca un resultado incorrecto, un resultado inesperado y/o se comporte de manera inconsistente, inesperada o involuntaria. Tal como se utiliza en el presente documento, la frase "falso positivo" es un resultado que indica incorrectamente que un problema detectado es un error tal como se define en el presente documento.

Tal como se explicará posteriormente, cada modelo en el componente de evaluación puede interactuar con el componente de almacenamiento y el componente de generación de características y actualizarse y entrenarse a medida que se disponga de nuevos datos. Por ejemplo, una vez que el modelo ha determinado una probabilidad de si un problema es un error, esa información se envía a un desarrollador. Aunque el término "probabilidad" se utiliza en el presente documento, los modelos en el componente de evaluación pueden configurarse para clasificar los problemas detectados o proporcionar alguna otra indicación de que se ha determinado si un problema es un error. Por ejemplo, a cada uno de los problemas se le puede asignar un puntaje (por ejemplo, un puntaje de confianza u otro puntaje). Si la puntuación está por encima (o por debajo) de un umbral, puede indicar si un problema es un error introducido por un desarrollador o algún otro problema intermitente.

El desarrollador revisa los problemas y determina si el problema es un error, proporciona una solución (por ejemplo, actualiza el código fuente) al error y proporciona esta información al sistema. Esta información puede utilizarse para actualizar los modelos de entrenamiento y/o el componente de generación de características para que sean más precisos a medida que transcurre el tiempo.

Estos y otros ejemplos se describirán con más detalle posteriormente con respecto a las figuras 1 a 4.

La figura 1 ilustra un sistema 100 para evaluar y clasificar los problemas que surgen en un sistema de *software* de acuerdo con un ejemplo. En los ejemplos, el sistema de *software* puede incluir varios módulos y componentes que operan juntos para realizar una función. Estos ejemplos incluyen sistemas operativos, aplicaciones (por ejemplo, aplicaciones de procesamiento de texto, aplicaciones de hoja de cálculo, navegadores de internet y similares), entornos de cómputo en la nube, etcétera.

Con el transcurso del tiempo, los desarrolladores pueden actualizar el sistema de *software* o varios componentes del sistema de *software*. A medida que se realizan las actualizaciones, pueden probarse los diversos componentes del sistema de *software* (por separado o con otros componentes de *software* y/o

hardware) para determinar si una actualización de un componente tendrá un impacto negativo en otro componente del sistema de *software* y/o el sistema de *software* en su conjunto.

En el sistema 100, el sistema de *software* que se probará, que incluye todos sus componentes y *hardware* asociado (por ejemplo, equipos de red, bases de datos, servidores, etcétera) puede ejecutarse y probarse en el laboratorio de automatización 105. Tal como se describió brevemente antes, el laboratorio de automatización puede probar continuamente los diversos componentes de *software* del sistema de *software* a medida que se construye un nuevo *software* y/o se modifican los componentes de *software* existentes.

En los ejemplos, el laboratorio de automatización incluye uno o más monitores 110. Los monitores 110 detectan si ha surgido un problema durante el proceso de prueba. En algunos casos, diferentes monitores 110 monitorizan diferentes partes del sistema de *software*. Por ejemplo, un primer monitor puede abarcar uno o más componentes de *software* del sistema de *software*, mientras un segundo monitor abarca uno o más componentes de *hardware* que están asociados o que de otro modo interactúan con el sistema de *software*. El laboratorio de automatización 105 puede configurarse para ejecutar continuamente varios casos de prueba. En otras implementaciones, el laboratorio de automatización 105 puede configurarse para ejecutar casos de prueba a pedido.

Si uno o más monitores 110 en el laboratorio de automatización detectan un problema, la información sin procesar sobre el problema 115 se transmite a un componente de suministro de datos 120. En los ejemplos, el componente de suministro de datos 120 se configura para manejar todos los problemas 115 que se generan o detectan de otro modo desde el laboratorio de automatización 105.

Una vez que se reciben los problemas 115, el componente de suministro de datos 120 transforma la información sobre el problema 115 en un esquema estructurado que facilita el almacenamiento. En algunos casos, la información sobre el problema 115 manejado por el componente de suministro de datos 120 incluye verificaciones de ejecución, volcados de memoria, etcétera. Además de esta información, el componente de suministro de datos 120 también puede recibir información sobre el estado actual del laboratorio de automatización 105.

A partir de esta información, el componente de suministro de datos 120 puede determinar el contexto en el cual ocurrió el problema 115. Por ejemplo, el componente de suministro de datos 120 puede determinar el número de compilación del código fuente que se está probando, la hora y fecha actuales, qué versión del sistema operativo estaba ejecutando una computadora principal y otra información similar.

El componente de suministro de datos 120 también puede configurarse para generar una propiedad distintiva de los diversos problemas 115 a través de una funcionalidad de discretización en intervalos. Una vez que se ha establecido la propiedad distintiva del problema 115, se compara con las propiedades distintivas de otros problemas 115 que ya se informaron al componente de suministro de datos 120. Si se determina que el problema actual 115 es un problema nuevo, se almacena información sobre el problema 115, junto con su propiedad distintiva. En algunos casos, la discretización en intervalos permite la generación de varias estadísticas temporales (por ejemplo, número de coincidencias semanales, mensuales y durante la vida efectiva) de varios problemas 115 que pueden estar presentes en el sistema

de *software*.

Si se determina que el problema 115 es un problema nuevo, el componente de suministro de datos 120 empuja la información sobre el problema 115 a un componente de análisis de problemas 125. El componente de análisis de problemas 125 analiza la información, determina una probabilidad de si el problema 115 es un error o un falso positivo y utiliza esa información para entrenar sus modelos de evaluación.

En algunos casos, el componente de análisis de problemas 125 incluye o está asociado con una capa de almacenamiento 130. La capa de almacenamiento 130 recibe información sobre el problema 115 del componente de suministro de datos 120. En algunos ejemplos, la capa de almacenamiento 130 puede ser extensible y editable. La capa de almacenamiento 130 también puede configurarse para almacenar comentarios del desarrollador sobre diversos problemas que se detectan, así como cualquier comentario recibido y asociado proporcionado por un desarrollador. La capa de almacenamiento 130 también puede permitir que varios desarrolladores y/o componentes del sistema 100 ejecuten consultas de alcance (por ejemplo, consultas sobre un subconjunto) sobre los datos almacenados.

El componente de análisis de problemas 125 también incluye un componente de generación de características 140. El componente de generación de características 140 incluye uno o más generadores de características 145. Cada generador de características 145 está configurado para generar un vector de características 150 que está asociado con un problema particular 115.

En algunos casos, puede haber muchas clases diferentes de problemas que existen en un sistema de *software*. Por ejemplo, uno o más errores pueden estar relacionados con diferentes componentes de *software* y/o uno o más equipos de desarrollo diferentes. Además, los diversos problemas 115 pueden exhibir comportamientos diferentes. En consecuencia, cada generador de características 145 en el componente de generación de características 140 puede configurarse específicamente para cada tipo de problema. Como tal, el componente de generación de características puede incluir muchos generadores de características diferentes 145.

Por ejemplo, cada generador de características 145 puede configurarse para buscar información relacionada con el problema particular 115. Si se encuentra información relacionada con un problema particular en la información de problema 135, se genera un vector de característica asociado con ese problema particular. Por ejemplo, un generador de características particular puede configurarse para contar cuántas veces se producen el "error" y la "memoria" en una traza de la pila. Por lo tanto, si el generador de características recibió una traza de la pila de "error de memoria en la ubicación de memoria 1", el vector de características generado por el generador de características sería "[1,2]" ya que la palabra "error" aparece una vez y la palabra "memoria" aparece dos veces. Sin embargo, si la traza de la pila está vacía o no está presente, el vector de característica resultante puede ser "[0,0]". Una vez que se genera el vector de características, esa información se pasa al componente de evaluación.

En algunos casos, cada generador de características 145 puede recuperar varios puntos de datos de la capa de almacenamiento 130. Cada generador de características 145 también puede realizar una o más transformaciones en los datos basados, al menos en parte, en la configuración del generador de

características 145.

Por ejemplo, si un generador de características 145 está configurado para buscar un tipo particular de problema o está asociado con un componente de *software* o componente de *hardware* particular, el generador de características 145 puede realizar un tipo específico de transformación en los datos. Ejemplos de tales transformaciones podrían ser la normalización, contando la aparición de ciertas palabras clave a partir de trazas de la pila, discretización en intervalos, o de otra manera extracción o generación de características a partir de los datos sin procesar disponibles en la capa de almacenamiento 130. El generador de características 145 utiliza la información transformada para generar un vector de características 150.

Una vez que se generó el vector de características 150, el vector de características 150 se transmite a un componente de evaluación 155. El componente de evaluación 155 incluye uno o más modelos 160. En algunos ejemplos, cada modelo 160 tiene una correspondencia uno a uno con un generador de características particular 145.

Por ejemplo, cada generador de características 145 es responsable de generar vectores de características para un modelo 160.

De esta manera, el sistema 100 puede estar compuesto por modelos específicos de dominio altamente precisos 160. Además, esto mejora la flexibilidad del sistema 100 ya que los diferentes generadores de características 145 pueden ser desarrollados y experimentados por diferentes equipos de ingeniería de forma independiente, puesto que leen desde la capa de almacenamiento 130.

Aunque se menciona una correspondencia uno a uno, un generador de características 145 puede proporcionar vectores de características 150 a muchos modelos diferentes 160. Del mismo modo, un solo modelo 160 puede recibir vectores de características 150 de varios generadores de características diferentes 145.

En los ejemplos descritos en el presente documento, el componente de evaluación 155 se utiliza para determinar una probabilidad o pronosticar de otro modo si el vector de característica recibido 150 está asociado con un error o si está asociado con un falso positivo. Por ejemplo, el modelo 160 puede configurarse para determinar la probabilidad de que cada problema sea un error o ruido intermitente, pronosticando si este problema pertenece a un equipo determinado, etcétera. En los ejemplos, puede depender de los desarrolladores que utilizan el sistema 100 saber qué clases de errores se originan en su laboratorio de automatización 105. Como tal, un desarrollador puede crear y/o actualizar sus propios modelos 160 y producir los generadores 145 de acuerdo con las necesidades individuales.

En algunos casos, puede surgir un problema en el cual un modelo 160 en el componente de evaluación 155 no está equipado para manejar un pronóstico con base en la información contenida en el vector de características 150. En tales casos, el sistema 100 puede notificar a un desarrollador y permitir la creación de otro modelo 160 y/o generador de características 145.

Cada modelo 160 implementado en el componente de evaluación 155 también es capaz de interactuar con la capa de almacenamiento 130 y el componente de generación de características 140. Como tal, cada modelo 160 puede actualizarse y reentrenarse a medida que se proporcionan diversas

actualizaciones y/o retroalimentaciones y a medida que se dispone de nuevos datos de entrenamiento.

En algunos casos, cada generador de características 145 y cada modelo 160 en el componente de evaluación 155 funcionan independientemente uno del otro. Como tal, cada generador de características 145 y/o cada modelo 160 pueden entrenarse en cualquier momento. Además, cada modelo 160 puede ser un tipo diferente de modelo. Por ejemplo, un modelo podría ser un clasificador de máquina de vectores de soporte (SVM, por sus siglas en inglés) que esté entrenado en lotes, mientras otro modelo podría ser un clasificador de k vecinos más próximos (KNN, por sus siglas en inglés) entrenado en línea.

Una vez que uno o más modelos 160 del componente de evaluación 155 analizan el vector de características 150, se genera un pronóstico 165 y se proporciona a un componente de cliente 170. El componente de cliente 170 puede incluir uno o más clientes 175. Cada uno de uno o más clientes 175 puede interactuar con los otros componentes descritos anteriormente.

Por ejemplo, un primer cliente puede configurarse para interactuar con un modelo específico y un generador de características, mientras un segundo cliente se configura para interactuar con un modelo y un generador de características diferentes. El primer cliente puede configurarse para examinar nuevos problemas (por ejemplo, problemas aún no inspeccionados por un desarrollador) y utiliza el modelo 160 para pronosticar la probabilidad de que el problema sea un falso positivo o un error. Este cliente 175 puede proporcionarle a un desarrollador una lista priorizada de los problemas activos que tienen más probabilidades de ser un error.

Del mismo modo, el segundo cliente puede configurarse para interactuar con la capa de almacenamiento 130 y buscar errores u otros problemas 115 que aún están pendientes. El segundo cliente puede solicitar a su generador de características asociado 145 crear un vector de características 150 y proporcionar ese vector de características 150 al modelo 160 asociado. Tal como se describió anteriormente, el modelo 160 determina una probabilidad de si el problema 115 detectado es un error o si es un falso positivo. Esta información también puede enviarse a un desarrollador tal como se describió anteriormente.

Cuando el desarrollador accede al sistema 100, el desarrollador puede elegir uno o más errores para investigar. Durante la investigación, el desarrollador puede determinar que el problema 115 detectado es un error y proporcionar una solución para el error. Esta información, mostrada como retroalimentación 180 en la figura 1, se transmite y almacena en la capa de almacenamiento 130. La retroalimentación 180 también puede proporcionarse a los diversos generadores de características 145 en el componente de generación de características 140 y/o los modelos 160 en el componente de evaluación.

En algunos casos, el desarrollador puede determinar que el problema 115 informado es un falso positivo. En estos casos, la retroalimentación 180 sobre el problema 115 (por ejemplo, la retroalimentación que indica que el problema 115 no fue un error provocado por parte del desarrollador) también se envía a la capa de almacenamiento 130.

En algunos casos, el cliente 175 también puede proporcionar la retroalimentación directamente al generador de características 145 y/o al modelo 160 con el cual está asociado el problema. Por ejemplo y tal como se describió anteriormente, cada generador de características 145 y/o modelo 160 puede

asociarse o configurarse de otro modo para manejar un problema particular 115. Por lo tanto, cuando se detecta el mismo problema, o un problema similar en un momento posterior, el generador de características 145 y/o el modelo 160 pueden estar mejor entrenados sobre cómo manejar el problema 115, así como determinar si el problema 115 detectado es un error o un falso positivo. En otros casos, la información sobre un problema particular puede almacenarse en la capa de almacenamiento 130. Si surge el mismo problema o similar, la información sobre el problema se compara con la información del problema en la capa de almacenamiento 130 y después se transmite a un flujo de evaluación y generación de características sin estado. En algunos casos, el flujo de evaluación y generación de características sin estado puede ser similar al flujo de operaciones descrito en el presente documento.

El siguiente es un ejemplo de cómo puede utilizarse el sistema 100 descrito anteriormente. Aunque se describen detalles, estos están destinados a ser solo ejemplos y el sistema 100 puede utilizarse en una serie de escenarios diferentes.

En este ejemplo, un equipo de desarrolladores puede desarrollar *software* que analice documentos. Un componente analítico del *software*, así como la prueba de este componente, involucra múltiples máquinas y piezas de *software* diferentes. Por ejemplo, un corpus de prueba de documentos puede alojarse en el servidor de almacenamiento. Un recolector puede ejecutarse hacia el servidor de almacenamiento y configurarse para extraer texto de los documentos. Una vez que se extrajo el texto, el texto puede transmitirse al componente de análisis. El sistema también puede incluir otro servidor que ejecuta un sistema de base de datos que almacena los resultados del componente de análisis.

En este ejemplo, el equipo de desarrollo de *software* utiliza el sistema 100 descrito anteriormente. Por ejemplo, cuando se detecta un problema en un laboratorio de automatización, la información sobre el problema se transmite a una capa de almacenamiento (por ejemplo, la capa de almacenamiento 130). Un generador de características (por ejemplo, el generador de características 145 del componente de generación de características 140) extrae los datos a partir de la información que se consideró efectiva para detectar falsos positivos. Por ejemplo, el generador de características puede extraer la información sobre el número total de veces que ocurrió el problema (por ejemplo, coincidencias en la vida efectiva), el número total de veces que cada problema ocurrió el último mes (por ejemplo, coincidencias en el último mes), el número de veces que cada problema ocurrió la última semana (por ejemplo, coincidencias en la última semana), el número de compilación en el cual ocurrió el problema por última vez y la resolución final del problema (por ejemplo, si el problema se solucionó o una determinación de que el problema no es reproducible).

Aunque se menciona información específica, esto es solo para fines de ejemplo. En consecuencia, un generador de características puede no extraer tanta información como se indica o puede extraer información adicional.

Una vez que se ha recopilado esta información, parte de la información puede normalizarse aún más. Por ejemplo, la cantidad de coincidencias de la última semana y la cantidad de visitas del último mes pueden dividirse por la cantidad total de visitas en la vida efectiva. Estos números pueden utilizarse para determinar tendencias temporales, como si un problema en particular ocurrió con más frecuencia la última

semana que en las semanas anteriores.

El componente de generación de características utiliza esta información para generar un vector de características y lo proporciona a un modelo (por ejemplo, el modelo 160) en un componente de evaluación (por ejemplo, el componente de evaluación 155). Tal como se abordó anteriormente, el modelo determina la probabilidad de si el problema es un error o un falso positivo. En este ejemplo, el resultado del modelo es que a cada nuevo problema (por ejemplo, problemas que previamente no se identificaron, investigaron y/o corrigieron por parte de un desarrollador), se le asigna una probabilidad de ser un falso positivo. El modelo presenta estos pronósticos a un cliente (por ejemplo, cliente 175). El cliente entonces puede presentar los nuevos problemas al equipo de desarrollo de *software*. En algunos casos, los problemas pueden ordenarse utilizando la probabilidad que se determinó por el modelo.

El desarrollador puede seleccionar un problema, investigarlo e identificarlo como un error y solucionarlo o identificarlo como falso positivo. Si el desarrollador determina que el problema es un error, el desarrollador corrige el error y un elemento de problema asociado con el problema detectado se actualiza en la capa de almacenamiento. Si el desarrollador determina que el problema no es un error (por ejemplo, el problema es un falso positivo), esa información también se retroalimenta a la capa de almacenamiento para que el sistema pueda aprender de ella en el futuro.

El siguiente es otro ejemplo de cómo puede funcionar el sistema descrito en el presente documento. En este ejemplo, un desarrollador en el equipo de desarrollo de *software* puede abrir o de otro modo acceder al cliente (o puede acceder directamente a la capa de almacenamiento) y consultar en la capa de almacenamiento los problemas que aún no se resolvieron (por ejemplo, errores que aún no se solucionaron, considerados falsos positivos o considerados no reproducibles). Estos problemas se envían al componente de generación de características, que genera un conjunto de vectores de características para los problemas.

Los vectores de características se alimentan a los componentes de evaluación de modelo, lo que genera una probabilidad de que el problema sea un falso positivo o un error. Los problemas pueden ordenarse de acuerdo con la probabilidad de ser problemas reales (o pueden ordenarse de alguna otra manera, como, por ejemplo, la mayor cantidad de tiempo que el problema ha estado presente, la cantidad de veces que ha surgido el problema, etcétera).

El desarrollador elige un problema (por ejemplo, un problema que tiene la mayor probabilidad de ser un error) y lo investiga. Si el desarrollador determina que el problema es un error, el desarrollador corrige el error. Un elemento de problema en la capa de almacenamiento se actualiza con la información de que el problema era un error y la solución que se realizó. Si el desarrollador determinó que el problema es un falso positivo, esa información también se retroalimenta a la capa de almacenamiento para que el sistema pueda aprender de ella en el futuro. El desarrollador puede seleccionar un nuevo problema y el ciclo se repite.

Los procesos descritos anteriormente pueden utilizarse para mejorar la precisión del sistema 100 utilizando tanto la intervención humana como el conocimiento con análisis computarizado. Como tal, los desarrolladores pueden aumentar la calidad de los datos de entrenamiento con cada problema que se

maneja. En algunos casos, los modelos dentro del sistema pueden entrenarse en diferentes momentos y con diferentes frecuencias. Por ejemplo, el entrenamiento de los modelos puede depender de la frecuencia con la cual se incorpore la retroalimentación en el modelo y también puede depender del tipo de modelo y la rapidez con la cual se solucionan los problemas. Por ejemplo, en un algoritmo en línea, cada nuevo elemento de retroalimentación podría incorporarse de inmediato. Por el contrario, un modelo que se crea en forma de computación por lotes podría regenerarse una vez por hora, una vez al día, una vez a la semana, etcétera.

En algunos casos, el sistema 100 descrito anteriormente puede utilizarse para concluir automáticamente problemas que se consideran de naturaleza intermitente o falsa. En algunos casos, esta determinación puede llevarse a cabo con base, al menos en parte, en una determinación de probabilidad. Por ejemplo, un cliente del sistema puede comparar la probabilidad de que cada problema sea un falso positivo con un umbral determinado. Si se supera el umbral, el cliente actualiza el elemento de problema en la capa de almacenamiento para indicar que no era reproducible. En algunos casos, la determinación puede basarse en una determinación previa realizada por un desarrollador.

En algunos casos, y para evitar que el sistema continúe evaluando problemas sin intervención humana, los problemas pueden etiquetarse de manera que no se incluyan en el conjunto de datos utilizado para entrenar nuevos modelos. Sin embargo, puesto que la representación de la capa de almacenamiento de los problemas puede editarse posteriormente, pueden crearse herramientas para permitir a los desarrolladores estar de acuerdo con las decisiones tomadas por el sistema y/o corregir cualquier error que se realizó por la determinación automática. En cualquier caso, la decisión de aceptar o corregir los datos puede utilizarse para entrenar los modelos y/o los generadores de características.

Aunque, el sistema se describió con respecto a un laboratorio de automatización, el sistema 100 descrito anteriormente podría integrarse en cualquier otro sistema. Asimismo, el sistema 100 podría observar otro sistema desde el exterior. En otro ejemplo más, podrían utilizarse múltiples sistemas (por ejemplo, un sistema integrado y un sistema externo) en un solo sistema de *software*.

Del mismo modo, el sistema 100 puede implementarse para ejecutarse como un cliente enriquecido que se ejecuta en una única computadora principal, como un sistema del servidor (por ejemplo, uno que se proporciona como interfaz web) o en cualquier combinación de estos (por ejemplo, con un servidor interno y un cliente más liviano frontal).

La figura 2 ilustra un método 200 para evaluar y clasificar los problemas que surgen en un sistema de *software* de acuerdo con un ejemplo. En algunos ejemplos, el método 200 puede utilizarse por un sistema 100 como el descrito anteriormente con respecto a la figura 1.

El método 200 comienza en la operación 210 en la cual los datos asociados con un problema detectado se detectan y se proporcionan al sistema. En este ejemplo, la operación 210 también puede utilizarse para determinar si el problema es un problema nuevo o si este problema se detectó previamente. La información sobre el problema puede proporcionarse al sistema por un componente de suministro de datos tal se describió anteriormente. Además, la información sobre el problema puede almacenarse en un componente de almacenamiento de datos.

Una vez que se reciban los datos sobre el problema, el flujo procede a la operación 220 y los datos se proporcionan a un componente de generación de características. El componente de generación de características analiza la información y, con base en el modelo al que transmitirá información, genera un vector de características que describe, identifica o se asocia con el problema. Por ejemplo, el componente de generación de características puede incluir uno o más generadores de características que se utilizan para determinar si una palabra en particular está presente en la información, la cantidad de veces que un error particular está presente en la información, etcétera. El generador de características entonces genera un vector de características tal como se describió anteriormente.

En la operación 230, el vector de características se proporciona a un componente de evaluación. En algunos ejemplos, el componente de evaluación tiene uno o más modelos que evalúan el vector de características y determina una probabilidad de que el vector de características identifique o se asocie con un error. El modelo puede configurarse para buscar un tipo específico de error. Como tal, el modelo puede recibir un vector de características que se generó específicamente para ese tipo de modelo.

Por lo tanto, múltiples modelos diferentes pueden analizar cada uno de los vectores de características que se crearon específicamente para ese modelo pero que se originan a partir del mismo problema. Es decir, la información del problema puede enviarse a múltiples generadores de características, cada uno de los cuales genera un vector de características específico para su modelo asociado. En otros ejemplos, el componente de generación de características puede utilizarse para determinar, con base en la información de problema recibida, qué generador de características debe utilizarse para crear el vector de características.

En la operación 240, la determinación de probabilidad se proporciona a un dispositivo cliente. En algunos casos, más de un problema puede estar presente en el cliente. En tales casos, el orden de los problemas informados puede enumerarse o clasificarse con base en ciertos criterios. Por ejemplo, los problemas pueden enumerarse en orden de probabilidad de que el problema detectado sea un error, que el problema detectado sea un falso positivo, el tiempo que el problema estuvo presente, etcétera.

En la operación 250, se determina si se recibió la retroalimentación para un problema particular. Por ejemplo, puede determinarse si un desarrollador seleccionó un problema particular, analizó el problema para confirmar si el problema es un error o un falso positivo y, si el problema es un error, proporcionó una solución al problema. Si no se recibió la retroalimentación, el flujo puede regresar a la operación 210 y las operaciones se repiten.

Sin embargo, si se seleccionó un problema y se solucionó o se determinó que el problema era un falso positivo (por ejemplo, por parte de un desarrollador), y se proporcionó retroalimentación, el flujo procede a la operación 260. En la operación 260, la retroalimentación puede utilizarse para entrenar los diversos componentes del sistema. Por ejemplo, la retroalimentación puede utilizarse para entrenar los modelos en el componente de evaluación y/o los diversos generadores de características en el componente de generación de características. La retroalimentación, junto con la información sobre el problema, también puede almacenarse en una capa de almacenamiento del sistema.

La figura 3 ilustra un método 300 para resolver automáticamente problemas que surgen en un

sistema de *software* de acuerdo con los ejemplos. En algunos casos, el método 300 puede utilizarse por el sistema 100 descrito anteriormente con respecto a la figura 1.

El método 300 comienza en la operación 310 en la cual se recibe una probabilidad (u otro indicador de clasificación y/o puntuación) sobre si un problema detectado es un error. En otras implementaciones, la probabilidad recibida en la operación 310 puede ser una probabilidad de que el problema sea un falso positivo. En cualquier caso, la probabilidad puede recibirse por un modelo asociado con un componente de evaluación.

El flujo entonces procede a la operación 320 en la cual la probabilidad que se recibió en la operación 310 se compara con un umbral de probabilidad. En algunos casos, el umbral de probabilidad puede establecerlo un desarrollador. En otros casos, el umbral de probabilidad puede ajustarse automáticamente por el sistema con base, al menos en parte, en las retroalimentaciones proporcionadas por parte del desarrollador. Por ejemplo, si el sistema determina que un problema es un error y entonces el desarrollador confirma esa determinación, el umbral de probabilidad puede modificarse en consecuencia (por ejemplo, disminuirse o aumentarse). Si el desarrollador tiene que corregir la determinación, el umbral de probabilidad también puede modificarse.

En algunos casos, cada tipo de problema puede asociarse con un umbral de probabilidad diferente. En tales casos, el umbral de probabilidad asociado con el problema particular puede ajustarse con base en las retroalimentaciones recibidas de parte del desarrollador. En otros casos, todos los problemas informados pueden estar asociados con el mismo umbral de probabilidad y ajustarse independientemente del tipo de problema informado y abordado automáticamente por el sistema.

El flujo entonces pasa a la operación 330 y se determina si la probabilidad excede el umbral de probabilidad. Si la probabilidad excede el umbral, el flujo procede a la operación 340 y el problema se informa al desarrollador de la manera descrita anteriormente con respecto a la figura 2. Entonces, el flujo procede a la operación 370 y cualquier retroalimentación de parte del desarrollador se proporciona al sistema y se utiliza para entrenar los modelos y/o los generadores de características tal como se describió anteriormente.

Sin embargo, si se determina que la probabilidad no supera el umbral (por ejemplo, se determina que el problema detectado no es un error), el sistema resuelve automáticamente el problema en la operación 350. Entonces, el flujo procede a la operación 360 y el problema se etiqueta o se marca de otro modo, lo cual indica que un desarrollador puede desear revisar la resolución del problema. Entonces, el flujo procede a la operación 370 y cualquier retroalimentación recibida (por ejemplo, retroalimentación positiva y/o negativa) de parte del desarrollador puede proporcionarse al sistema y utilizarse para entrenar a los diversos componentes del sistema, tal como se describió anteriormente.

La figura 4 ilustra un método 400 para recibir retroalimentaciones para un problema detectado. En algunos casos, el método 400 puede utilizarse por el sistema 100 descrito anteriormente. Además, el método 400 puede utilizarse por un dispositivo cliente que proporciona información sobre problemas a un desarrollador.

El método 400 comienza en la operación 410 en la cual el dispositivo cliente recibe una lista

ordenada de información de problema. En algunos casos, la lista ordenada de información de problema se recibe por un componente de evaluación, como, por ejemplo, el componente de evaluación 155. En otros casos, la lista de información de problema puede recibirse a partir de la información de problema almacenada en un dispositivo de almacenamiento asociado con el sistema.

La lista puede ordenarse de acuerdo con cualquier número de factores. Por ejemplo, la lista podría ordenarse con base en la probabilidad de que un problema sea un error, una probabilidad de que el problema sea un falso positivo, la fecha en la cual se detectó el problema, la frecuencia de la detección, etcétera. En otros casos, la lista de problemas puede presentarse en un orden aleatorio, un orden de primeras entradas, primeras salidas, orden de últimas entradas, últimas salidas, etcétera.

Una vez que se recibió la información de problema, el flujo procede a la operación 420 y la lista de problemas se proporciona a un desarrollador. Una vez recibida, el dispositivo cliente puede recibir una selección de un problema particular (operación 430) y, en respuesta a la selección, proporcionar información adicional (operación 440) sobre el problema. Por ejemplo, el dispositivo cliente puede proporcionar la fecha y hora del problema, en cuál módulo o componente se detectó el problema, el área del código fuente en la cual se origina el problema, etcétera.

Entonces, el flujo procede a la operación 450 en la cual se reciben las retroalimentaciones sobre el problema. En algunos casos, las retroalimentaciones pueden proporcionarse por parte de un desarrollador una vez que el desarrollador investigue el problema. Por ejemplo, las retroalimentaciones pueden confirmar que el problema era un error, indicar que el problema no era un error según lo indicado por el sistema, confirmar que el problema era un falso positivo o indicar que el problema no era un falso positivo según lo indicado por el sistema. En algunos casos, las retroalimentaciones también pueden incluir correcciones del código fuente, actualizaciones del código fuente u otras retroalimentaciones sobre cómo se resolvió el problema.

Una vez que se recibe la retroalimentación, el flujo procede a la operación 460 y la retroalimentación se proporciona al sistema. En algunos casos, la retroalimentación se utiliza para entrenar a los diversos componentes del sistema. Por ejemplo, la retroalimentación puede utilizarse para entrenar o actualizar uno o más modelos de un componente de evaluación, crear uno o más modelos de un componente de evaluación, entrenar o actualizar uno o más generadores de características de un componente de generación de características y/o crear uno o más generadores de características de un componente de generación de características. En algunos casos, las retroalimentaciones también pueden almacenarse en un dispositivo de almacenamiento del sistema.

Una vez que se reciba la retroalimentación, el método o partes del método pueden repetirse. Por ejemplo, tal como se muestra en la figura 4, una vez que se recibe la retroalimentación, el flujo puede volver a la operación 430 en la cual un desarrollador selecciona otro problema en la lista y el proceso se repite.

En algunos casos, pueden detectarse o proporcionarse problemas adicionales al dispositivo cliente mientras el desarrollador está investigando un problema seleccionado. En tales casos, la lista ordenada de información de problemas puede actualizarse continuamente. En otros casos, la lista de información de problemas puede proporcionarse al dispositivo cliente periódicamente, a petición o cualquier combinación

de los mismos.

Las figuras 5 a 8 y las descripciones asociadas proporcionan un análisis de una variedad de entornos operativos en los cuales pueden ponerse en práctica los aspectos de la divulgación. Sin embargo, los dispositivos y sistemas ilustrados y abordados con respecto a las figuras 5 a 8 son para fines de ejemplo e ilustración y no limitan una gran cantidad de configuraciones de dispositivos electrónicos que pueden utilizarse para poner en práctica los aspectos de la divulgación, tal como se describe en el presente documento.

La figura 5 es un diagrama de bloques que ilustra los componentes físicos (por ejemplo, hardware) de un dispositivo electrónico 500 con el cual se pueden ponerse en práctica los aspectos de la divulgación. Los componentes del dispositivo electrónico 500 que se describen posteriormente pueden tener instrucciones ejecutables por computadora para identificar automáticamente problemas en un paquete de *software* tal como se describió anteriormente.

En una configuración básica, el dispositivo electrónico 500 puede incluir al menos una unidad de procesamiento 510 y una memoria del sistema 515. Dependiendo de la configuración y del tipo de dispositivo electrónico, la memoria de sistema 515 puede comprender, entre otros, almacenamiento volátil (por ejemplo, memoria de acceso aleatorio (RAM, por sus siglas en inglés)), almacenamiento no volátil (por ejemplo, memoria de solo lectura (ROM, por sus siglas en inglés)), memoria flash o cualquier combinación de tales memorias. La memoria de sistema 515 puede incluir un sistema operativo 525 y uno o más módulos de programa 520 adecuados para identificar diversos problemas, tal como se describe en el presente documento.

El sistema operativo 525, por ejemplo, puede ser adecuado para controlar la operación del dispositivo electrónico 500. Además, los ejemplos de la divulgación pueden ponerse en práctica junto con una biblioteca de gráficos, otros sistemas operativos o cualquier otro programa de aplicación y no se limita a ninguna aplicación o sistema particular. Esta configuración básica se ilustra en la figura 5 por esos componentes dentro de una línea discontinua 530.

El dispositivo electrónico 500 puede tener características o funcionalidades adicionales. Por ejemplo, el dispositivo electrónico 500 también puede incluir dispositivos de almacenamiento de datos adicionales (removibles y/o no removibles) como, por ejemplo, discos magnéticos, discos ópticos o cintas. Tal almacenamiento adicional se ilustra en la figura 5 por un dispositivo de almacenamiento removable 535 y un dispositivo de almacenamiento no removable 540.

Tal como se indicó anteriormente, pueden almacenarse varios módulos de programa y archivos de datos en la memoria del sistema 515. Mientras se ejecuta en la unidad de procesamiento 510, los módulos de programa 520 (por ejemplo, un módulo de generación de características 505 o componente, un módulo de evaluación 507 o componente, etcétera, que puede comprender el sistema 100) pueden realizar procesos que incluyen, entre otros, los aspectos, tal como se describe en el presente documento.

Además, pueden ponerse en práctica los ejemplos de la divulgación en un circuito eléctrico que comprende elementos electrónicos discretos, chips electrónicos integrados o empaquetados que contienen puertas lógicas, un circuito que utiliza un microprocesador, o en un solo chip que contiene elementos

electrónicos o microprocesadores. Por ejemplo, los ejemplos de la divulgación pueden ponerse en práctica a través de un sistema en un chip (SOC, por sus siglas en inglés) donde cada uno o muchos de los componentes ilustrados en la figura 5 pueden integrarse en un solo circuito integrado. Tal dispositivo SOC puede incluir una o más unidades de procesamiento, unidades gráficas, unidades de comunicaciones, unidades de virtualización de sistema y varias funciones de aplicación, todas ellas integradas (o "grabadas") en el sustrato del chip como un solo circuito integrado.

Cuando se opera a través de un SOC, la funcionalidad, descrita en el presente documento, con respecto a la capacidad del cliente para cambiar de protocolo, puede operarse a través de la lógica específica de la aplicación integrada con otros componentes del dispositivo electrónico 500 en el circuito integrado único (chip). Los ejemplos de la divulgación también pueden ponerse en práctica utilizando otras tecnologías capaces de realizar operaciones lógicas como, por ejemplo, AND, OR y NOT, que incluyen, entre otras, tecnologías mecánicas, ópticas, fluidicas y cuánticas. Además, los ejemplos de la divulgación pueden ponerse en práctica dentro de una computadora de uso general o en cualquier otro circuito o sistema.

El dispositivo electrónico 500 también puede tener uno o más dispositivos de entrada 545 como un teclado, un panel táctil, un ratón, un bolígrafo, un dispositivo de entrada de sonido o voz, un dispositivo de entrada táctil, aplicación de fuerza y/o deslizamiento, etcétera. También pueden incluirse los dispositivos de salida 550, como una pantalla, altavoces, una impresora, etcétera. Los dispositivos mencionados anteriormente son ejemplos y pueden utilizarse otros. El dispositivo electrónico 500 puede incluir una o más conexiones de comunicación 555 que permiten comunicaciones con otros dispositivos electrónicos 560. Los ejemplos de conexiones de comunicación adecuadas 555 incluyen, entre otros, transmisor, receptor y/o circuito transceptor de radiofrecuencia (RF, por sus siglas en inglés); bus universal en serie (USB, por sus siglas en inglés), puertos paralelos y/o seriales.

El término "medio legible por computadora" tal como se utiliza en el presente documento puede incluir medios de almacenamiento por computadora. Los medios de almacenamiento por computadora pueden incluir medios volátiles y no volátiles, removibles y no removibles implementados en cualquier método o tecnología para el almacenamiento de información, como instrucciones legibles por computadora, estructuras de datos o módulos de programa.

La memoria de sistema 515, el dispositivo de almacenamiento removible 535 y el dispositivo de almacenamiento no removible 540 son todos ejemplos de medios de almacenamiento por computadora (por ejemplo, almacenamiento de memoria). Los medios de almacenamiento por computadora pueden incluir RAM, ROM, memoria de solo lectura eléctricamente borrable (EEPROM, por sus siglas en inglés), memoria flash u otra tecnología de memoria, disco compacto con memoria de solo lectura (CD-ROM, por sus siglas en inglés), discos versátiles digitales (DVD, por sus siglas en inglés) u otro almacenamiento óptico, casetes magnéticos, cinta magnética, almacenamiento en disco magnético u otros dispositivos de almacenamiento magnético, o cualquier otro artículo de fabricación que pueda utilizarse para almacenar información y al que pueda acceder el dispositivo electrónico 500. Cualquiera de estos medios de almacenamiento por computadora puede formar parte del dispositivo electrónico 500. Los medios de

almacenamiento por computadora no incluyen una onda portadora u otra señal de datos propagada o modulada.

Los medios de comunicación pueden incorporarse mediante instrucciones legibles por computadora, estructuras de datos, módulos de programa u otros datos en una señal de datos modulada, como una onda portadora u otro mecanismo de transporte, e incluyen cualquier medio de suministro de información. El término "señal de datos modulados" puede describir una señal que tiene una o más características establecidas o cambiadas de tal manera que codifique la información en la señal. A modo de ejemplo, y sin limitación, los medios de comunicación pueden incluir medios alámbricos como una red alámbrica o conexión directa, y medios inalámbricos como acústicos, radiofrecuencia (RF), infrarrojos y otros medios inalámbricos.

Las figuras 6A y 6B ilustran un dispositivo electrónico móvil 600, por ejemplo, un teléfono móvil, un teléfono inteligente, una computadora portátil (como un reloj inteligente), una tableta, una computadora portátil y similares, con los cuales los ejemplos de la divulgación pueden ponerse en práctica. Con referencia a la figura 6A, se ilustra un aspecto de un dispositivo electrónico móvil 600 para implementar los aspectos.

En una configuración básica, el dispositivo electrónico móvil 600 es una computadora portátil que tiene elementos de entrada y elementos de salida. El dispositivo electrónico móvil 600 comúnmente incluye una pantalla 605 y uno o más botones de entrada 610 que permiten al usuario ingresar información en el dispositivo electrónico móvil 600. La pantalla 605 del dispositivo electrónico móvil 600 también puede funcionar como un dispositivo de entrada (por ejemplo, una pantalla que acepta entrada táctil y/o aplicación de fuerza).

Si se incluye, un elemento de entrada lateral opcional 615 permite una entrada adicional del usuario. El elemento de entrada lateral 615 puede ser un interruptor giratorio, un botón o cualquier otro tipo de elemento de entrada manual. En los aspectos alternativos, el dispositivo electrónico móvil 600 puede incorporar más o menos elementos de entrada. Por ejemplo, la pantalla 605 puede no ser una pantalla táctil en algunos ejemplos. En otra modalidad alternativa más, el dispositivo electrónico móvil 600 es un sistema de teléfono portátil, tal como un teléfono celular. El dispositivo electrónico móvil 600 también puede incluir un teclado opcional 635. El teclado opcional 635 puede ser un teclado físico o un teclado por "software" generado en la pantalla táctil.

En varios ejemplos, los elementos de salida incluyen la pantalla 605 para mostrar una interfaz gráfica de usuario (GUI, por sus siglas en inglés), un indicador visual 620 (por ejemplo, un diodo emisor de luz), y/o un transductor de audio 625 (por ejemplo, un altavoz). En algunos aspectos, el dispositivo electrónico móvil 600 incorpora un transductor de vibración para proporcionar al usuario retroalimentación táctil. En otro aspecto más, el dispositivo electrónico móvil 600 incorpora puertos de entrada y/o salida, como una entrada de audio (por ejemplo, un conector de micrófono), una salida de audio (por ejemplo, un conector de auriculares) y una salida de video (por ejemplo, un puerto de interfaz multimedia de alta definición (HDMI, por sus siglas en inglés)) para enviar o recibir señales desde un dispositivo externo.

La figura 6B es un diagrama de bloques que ilustra la arquitectura de un aspecto de un dispositivo

electrónico móvil 600. Es decir, el dispositivo electrónico móvil 600 puede incorporar un sistema (por ejemplo, una arquitectura) 640 para implementar algunos aspectos. En una modalidad, el sistema 640 se implementa como un "teléfono inteligente" capaz de ejecutar una o más aplicaciones (por ejemplo, navegador, correo electrónico, calendario, administradores de contactos, clientes de mensajería, juegos, clientes/reproductores de medios, selección de contenido y aplicaciones de compartición, etcétera). En algunos aspectos, el sistema 640 está integrado como un dispositivo electrónico, como un asistente digital personal integrado (PDA, por sus siglas en inglés) y un teléfono inalámbrico.

Uno o más programas de aplicación 650 pueden cargarse en la memoria 645 y ejecutarse en o en asociación con el sistema operativo 655. Los ejemplos de los programas de aplicación incluyen programas de marcado telefónico, programas de correo electrónico, programas de gestión de información personal (PIM, por sus siglas en inglés), programas de procesamiento de texto, programas de hojas de cálculo, programas de navegador de internet, programas de mensajería, etcétera.

El sistema 640 también incluye un área de almacenamiento no volátil 660 dentro de la memoria 645. El área de almacenamiento no volátil 660 puede utilizarse para almacenar información persistente que no debería perderse si el sistema 640 se apaga.

Los programas de aplicación 650 pueden utilizar y almacenar información en el área de almacenamiento no volátil 660, como correo electrónico u otros mensajes utilizados por una aplicación de correo electrónico, y similares. Una aplicación de sincronización (no mostrada) también reside en el sistema 640 y está programada para interactuar con una aplicación de sincronización correspondiente residente en una computadora principal para mantener la información almacenada en el área de almacenamiento no volátil 660 sincronizada con la información correspondiente almacenada en la computadora principal.

El sistema 640 tiene una fuente de alimentación 665, que puede implementarse como una o más baterías. La fuente de alimentación 665 puede incluir, además, una fuente de alimentación externa, como un adaptador de AC o una base de conexión con alimentación que complementa o recarga las baterías.

El sistema 640 también puede incluir una capa de interfaz de radio 670 que realiza la función de transmitir y recibir comunicaciones de radiofrecuencia. La capa de interfaz de radio 670 facilita la conectividad inalámbrica entre el sistema 640 y el "mundo exterior", a través de un operador de comunicaciones o proveedor de servicios. Las transmisiones hacia y desde la capa de interfaz de radio 670 se realizan bajo el control del sistema operativo 655. Dicho de otro modo, las comunicaciones recibidas por la capa de interfaz de radio 670 pueden difundirse a los programas de aplicación 650 a través del sistema operativo 655, y viceversa.

El indicador visual 620 puede utilizarse para proporcionar notificaciones visuales, y/o una interfaz de audio 675 puede utilizarse para producir notificaciones audibles a través de un transductor de audio (por ejemplo, el transductor de audio 625 ilustrado en la figura 6A). En la modalidad ilustrada, el indicador visual 620 es un diodo emisor de luz (LED, por sus siglas en inglés) y el transductor de audio 625 puede ser un altavoz. Estos dispositivos pueden conectarse directamente a la fuente de alimentación 665 para que, cuando se activen, permanezcan encendidos durante un tiempo determinado por el mecanismo de notificación a pesar de que el procesador 685 y otros componentes podrían apagarse para conservar la

energía de la batería. El LED puede programarse para que permanezca encendido indefinidamente hasta que el usuario tome medidas para indicar el estado de encendido del dispositivo.

La interfaz de audio 675 se utiliza para proporcionar señales audibles y recibir señales audibles de parte del usuario (por ejemplo, entrada de voz tal como se describió anteriormente). Por ejemplo, además, de conectarse al transductor de audio 625, la interfaz de audio 675 también puede conectarse a un micrófono para recibir una entrada audible, como para facilitar una conversación telefónica. De acuerdo con los ejemplos de la presente divulgación, el micrófono también puede servir como un sensor de audio para facilitar el control de las notificaciones, tal como se describirá posteriormente.

El sistema 640 puede incluir, además, una interfaz de video 680 que permite la operación del dispositivo periférico 630 (por ejemplo, cámara integrada) para grabar imágenes fijas, transmisión de video y similares.

Un dispositivo electrónico móvil 600 que implementa el sistema 640 puede tener características o funcionalidades adicionales. Por ejemplo, el dispositivo electrónico móvil 600 también puede incluir dispositivos de almacenamiento de datos adicionales (removibles y/o no removibles) como discos magnéticos, discos ópticos o cintas. Tal almacenamiento adicional se ilustra en la figura 6B por el área de almacenamiento no volátil 660.

Los datos/información generados o capturados por el dispositivo electrónico móvil 600 y almacenados a través del sistema 640 pueden almacenarse localmente en el dispositivo electrónico móvil 600, tal como se describió anteriormente, o los datos pueden almacenarse en cualquier cantidad de medios de almacenamiento a los que pueda acceder el dispositivo a través de la capa de interfaz de radio 670 o mediante una conexión por cable entre el dispositivo electrónico móvil 600 y un dispositivo electrónico separado asociado con el dispositivo electrónico móvil 600, por ejemplo, una computadora servidor en una red de cómputo distribuida, como internet. Tal como debería apreciarse, puede accederse a tales datos/información a través del dispositivo electrónico móvil 600 a través de la capa de interfaz de radio 670 o a través de una red de cómputo distribuida. De manera similar, tales datos/información pueden transferirse fácilmente entre dispositivos electrónicos para su almacenamiento y uso de acuerdo con medios de transferencia y almacenamiento de datos/información bien conocidos, incluidos el correo electrónico y los sistemas colaborativos de intercambio de datos/información.

Tal como debe apreciarse, la figura 6A y la figura 6B se describen con el propósito de ilustrar los métodos y sistemas actuales y no se pretende limitar la divulgación a una secuencia particular de etapas o una combinación particular de componentes de *hardware* o *software*.

La figura 7 ilustra un aspecto de la arquitectura de un sistema 700 para informar automáticamente problemas detectados con sistemas de *software* como los descritos en el presente documento. El sistema 700 puede incluir un dispositivo electrónico general 710 (por ejemplo, computadora personal), dispositivo electrónico tipo tableta 715 o dispositivo electrónico móvil 720, tal como se describió anteriormente. Cada uno de estos dispositivos puede utilizarse para recibir problemas informados y/o proporcionar retroalimentaciones 725 tal como se describe.

En algunos aspectos, cada uno de los dispositivos electrónicos generales 710 (por ejemplo,

computadora personal), dispositivo electrónico tipo tableta 715 o dispositivo electrónico móvil 720 puede recibir varios otros tipos de información o contenido que se almacena o transmite desde un servicio de directorio 745, un portal web 750, servicios de buzón 755, almacenamientos de mensajería instantánea 760 o servicios de red social 765.

En algunos aspectos, los problemas y/o retroalimentaciones 725 pueden proporcionarse, a través de la red 730, al sistema de identificación de problemas 735 (o componentes del sistema de identificación de problemas) en un servidor 705 (aunque el sistema de identificación de problemas 735 también puede ser un sistema local).

A modo de ejemplo, los aspectos descritos anteriormente pueden realizarse en un dispositivo electrónico general 710 (por ejemplo, computadora personal), un dispositivo electrónico tipo tableta 715 y/o un dispositivo electrónico móvil 720 (por ejemplo, un teléfono inteligente). Cualquiera de estos ejemplos de dispositivos electrónicos puede obtener contenido o proporcionar datos al almacenamiento 740.

Tal como debe apreciarse, la figura 7 se describe con el fin de ilustrar los métodos y sistemas actuales y no pretende limitar la divulgación a una secuencia particular de etapas o una combinación particular de componentes de *hardware* o *software*.

La figura 8 ilustra un ejemplo de dispositivo electrónico tipo tableta 800 que puede ejecutar uno o más aspectos descritos en el presente documento. Además, los aspectos y funcionalidades descritos en el presente documento pueden operar a través de sistemas distribuidos (por ejemplo, sistemas de cómputo en la nube), donde la funcionalidad de la aplicación, la memoria, el almacenamiento y la recuperación de datos y varias funciones de procesamiento pueden operarse de forma remota entre sí a través de una red de cómputo distribuida, como internet o una intranet. Las interfaces de usuario y la información de varios tipos pueden mostrarse a través de pantallas de dispositivos electrónicos integrados o mediante unidades de presentación remotas asociadas con uno o más dispositivos electrónicos. Por ejemplo, las interfaces de usuario y la información de varios tipos pueden mostrarse e interactuar con ellas en una superficie de pared sobre la cual se proyectan las interfaces de usuario y la información de varios tipos. La interacción con la multitud de sistemas de cómputo con los que pueden ponerse en práctica los ejemplos de la invención incluye, entrada de pulsación de tecla, entrada de pantalla táctil, entrada de voz u otra entrada de audio, entrada de gestos donde un dispositivo electrónico asociado está equipado con funcionalidad de detección (por ejemplo, cámara) para capturar e interpretar gestos de usuario para controlar la funcionalidad del dispositivo electrónico, y similares.

Tal como debe apreciarse, en el presente documento la figura 8 se describe con el fin de ilustrar los métodos y sistemas actuales y no pretende limitar la divulgación a una secuencia particular de etapas o una combinación particular de componentes de *hardware* o *software*.

Tal como se describió anteriormente, se describe en el presente documento un método para determinar automáticamente si un problema detectado en un sistema de cómputo es un error, que comprende: recibir información sobre un problema detectado; almacenar la información sobre el problema detectado en un dispositivo de almacenamiento; generar un vector de características para el problema detectado; analizar el vector de características para determinar una probabilidad de que el problema

detectado sea un error, en el cual el análisis se basa, al menos en parte, en el vector de características; y cuando la probabilidad está por encima de un umbral, informar el problema detectado a un dispositivo cliente. En otros ejemplos, el método incluye recibir una solución al problema detectado. En otros ejemplos, el método incluye actualizar el análisis con base, al menos en parte, en la solución recibida para el problema detectado. En otros ejemplos, el método incluye asociar la solución con el problema detectado; y almacenar la solución en el dispositivo de almacenamiento. En otros ejemplos, el método incluye priorizar el problema detectado con respecto a uno o más problemas detectados adicionales. En otros ejemplos, el método incluye generar un vector de características que comprende realizar una o más transformaciones en la información sobre el problema detectado. En otros ejemplos, una o más transformaciones se seleccionan a partir de un grupo que comprende: normalización, recuento de aparición de palabras clave de una traza de la pila y discretización en intervalos.

También se describe un sistema que comprende: al menos una unidad de procesamiento; y una memoria que almacena instrucciones ejecutables por computadora que, cuando se ejecutan por al menos una unidad de procesamiento, hacen que el sistema realice un método para detectar ocurrencias de problemas en un sistema de *software*, que comprende: recibir información sobre un problema detectado; comparar una propiedad distintiva del problema detectado con una o más propiedades distintivas de problemas detectados previamente para determinar si el problema detectado es nuevo; cuando se determina que el problema detectado es nuevo: almacenar la información sobre el problema detectado en un dispositivo de almacenamiento; proporcionar la información sobre el problema detectado a un componente de generación de características que genera un vector de características para el problema detectado; proporcionar el vector de características a un componente de evaluación que analiza el vector de características para determinar una probabilidad de que el problema detectado sea un error; y proporcionar una lista ordenada de problemas detectados. En otros ejemplos, el componente de evaluación se asocia con uno o más modelos y donde cada uno de los modelos se configura para evaluar un vector de características recibido con base, al menos en parte, en la información contenida en el vector de características. En otros ejemplos, la memoria almacena instrucciones para recibir una solución al problema detectado. En otros ejemplos, la memoria almacena instrucciones para asociar la solución con el problema detectado. En otros ejemplos, la memoria almacena instrucciones para actualizar uno o más modelos del componente de evaluación con base, al menos en parte, en la solución. En otros ejemplos, un orden de la lista ordenada se basa en la probabilidad determinada. En otros ejemplos, la memoria almacena instrucciones para asignar una prioridad al problema detectado en vista de los problemas detectados previamente. En otros ejemplos, la memoria almacena instrucciones para almacenar información sobre el problema detectado en el dispositivo de almacenamiento cuando se determina que el problema detectado no es un error.

También se describe un medio de almacenamiento legible por computadora que almacena instrucciones ejecutables por computadora que, cuando se ejecutan por una unidad de procesamiento, hacen que la unidad de procesamiento realice un método, que comprende: generar un vector de características para un problema detectado utilizando información asociada con el problema detectado;

proporcionar el vector de características a uno o más modelos de un componente de evaluación; haciendo que uno o más modelos del componente de evaluación analicen el vector de características para determinar la probabilidad de que el problema detectado sea un error; informar la ocurrencia del problema detectado; recibir una solución al problema detectado; y actualizar al menos uno de uno o más modelos utilizando la solución al problema detectado. En otros ejemplos, la memoria almacena instrucciones para almacenar la solución en un dispositivo de almacenamiento. En otros ejemplos, la memoria almacena instrucciones para asociar la solución con el problema detectado. En otros ejemplos, la memoria almacena instrucciones para almacenar la información sobre el problema detectado. En otros ejemplos, la memoria almacena instrucciones para determinar qué modelo de uno o más modelos se configura para analizar el vector de características.

La descripción y la ilustración de uno o más aspectos proporcionados en esta solicitud no pretenden limitar o restringir de ninguna manera el alcance de la divulgación tal como se reclama. Los aspectos, ejemplos y detalles proporcionados en esta solicitud se consideran suficientes para transmitir la materia objeto y permitir que otros realicen y utilicen el mejor modo de la divulgación reclamada. La divulgación reclamada no debe interpretarse como limitada a ningún aspecto, ejemplo o detalle proporcionado en esta solicitud. Independientemente de si se muestra y describe en combinación o por separado, las diversas características (tanto estructurales como metodológicas) están destinadas a incluirse u omitirse selectivamente para producir una modalidad con un conjunto particular de características. Un experto en la materia al tener la descripción y la ilustración de la presente solicitud puede concebir variaciones, modificaciones y aspectos alternativos que se encuentran dentro de la esencia de los aspectos más amplios del concepto inventivo general incorporado en esta solicitud los cuales no se apartan del alcance más amplio de la divulgación reclamada.