INTRODUCCION

El presente trabajo está basado en los principales métodos de prueba de software. Describe los objetivos que se persiguen al realizar la prueba y los principios de la prueba.

Un producto puede probarse de acuerdo al conocimiento de la función específica para la que fue diseñado el producto (caja negra). Con la aplicación de esta técnica se adquieren pruebas que disminuyen el numero de casos de prueba.

La prueba del método de la caja de cristal frecuentemente es más eficaz para descubrir errores debido a que los programas sutiles ocurren en la interfaz de sub−programas.

La prueba de Pandora se utiliza muy a menudo.

Métodos de prueba del Software

Las pruebas son de gran importancia en la garantía de la calidad del software.

Los objetivos principales de realizar una prueba son:

- *Detectar un error*
- *Tener un buen caso de prueba*
- *Descubrir un error no descubierto antes.*

Los métodos de prueba del software tienen el objetivo de diseñar pruebas que descubran diferentes tipos de errores con menor tiempo y esfuerzo.

Principios de la prueba:

- *Hacer un seguimiento de las pruebas hasta los requisitos del cliente*
- *Plantear y diseñar las pruebas antes de generar ningún código*
- *El 80% de todos los errores se centran en solo en el 20% de los módulos*
- *Empezar las pruebas en módulos individuales y avanzar hasta probar el sistema entero.*
- *No son posibles las pruebas exhaustivas*
- *Deben realizarse por un equipo independiente al equipo de desarrollo*

Un software fácil de probar tiene la siguientes características:

- *Operatividad*
- *Objetividad*
- *Controlabilidad*
- *Capacidad de descomposición*
- *Simplicidad*
- *Estabilidad*
- *Facilidad de comprensión*

Atributos de una buena prueba

- *Más alta probabilidad de encontrar un error.*
- *No debe ser redundante*

- *No debería ser ni demasiado sencilla ni demasiado compleja*

DISEÑOS DE CASOS DE PRUEBA

Prueba de Caja Negra

Los datos de prueba se escogerán atendiendo a las especificaciones del problema, sin importar los detalles internos del programa, a fin de verificar que el programa corra bien.

Criterios mínimos que guiarán al escoger los datos de prueba:

- Valores Fáciles: *El programa se depurará con datos de fácil comprobabilidad.*
- Valores típicos realistas: *se ensayará un programa con datos seleccionados para que representen como se aplicará. Los Datos han de ser sencillos, de modo que los resultados sean verificables en forma manual.*
- Valores extremos
- Valores ilegales: *Cuando en un programa entra basura, su salida habrá de ser un mensaje de error adecuado. Es preferible que el programa ofrezca indicación de errores en la entrada y que realice cálculos que sigan siendo factibles luego de desechar la entrada equivocada.*

El método de la caja negra se centra en los requisitos fundamentales del software y permite obtener entradas que prueben todos los requisitos funcionales del programa. Con este equipo de pruebas se intenta encontrar:

- *Funciones incorrectas o ausentes.*
- *Errores de interfaz.*
- *Errores en estructuras de datos o en accesos a la bases de datos externas*
- *Errores de rendimiento.*
- *Errores de inicialización y terminación.*

Con la aplicación de esa técnica se obtiene un conjunto de pruebas que:

- *Reduce el numero de casos de pruebas y nos dicen algo sobre la presencia o ausencia de errores.*

Partición equivalente:

Una partición equivalente es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos. El diseño de casos de prueba para la partición equivalente se basa en la evaluación de las clases de equivalencia.

Análisis de valores limite:

Nos lleva a elegir las pruebas que nos ejecuten los valores límite, con esta técnica se complementa la partición equivalente.

Prueba de comparación:

Esta técnica consiste en la comparación de salidas de un mismo software pero de sus diferentes versiones.

Prueba de la Caja de Cristal

Principia con la observación de que un programa difícilmente puede considerarse como probado por completo si su código contiene partes que nunca han sido ejecutadas. Este método analiza la estructura lógica del programa y, para cada alternativa que puede presentarse, los datos de prueba ideados conducirán

a ella. Se procura escoger los que verifiquen cada posibilidad en las proposiciones ***case****, las cláusulas de cada proposición* ***if*** *y la condición de terminación de cada ciclo.*

En un programa extenso, este método es impráctico, pero en un modulo pequeño constituye un excelente medio de prueba y depuración.

En una prueba que se vale del método de la caja de cristal, se tornan patentes las ventajas de un diseño de programa modular.

Un buen criterio de prueba para proyectos extensos consiste en aplicar los métodos de la caja de cristal a cada módulo pequeño conforme se escriba; luego se usan esos datos en las secciones más amplias del programa una vez terminadas.

Prueba de la Caja de Pandora:

Consiste en abternerse de realizar pruebas de depurar bastante bien un proyecto; se deja al cliente que lo ensaye y acepte. El resultado es una bomba de tiempo.

CONCLUSION

Después de finalizar esta practica (trabajo de investigación) he comprendido que con las pruebas no puede asegurarse que no existen errores sólo pueden mostrar que existen defectos en el software.

La mayoría de los usuarios de programas extensos no se interesan en los detalles del funcionamiento del programa, lo que buscan son las respuestas.

En una prueba que se vale del método de la caja de cristal se tornan patentes las ventajas de una diseño de programa modular.

4