PROBA MOTAK.KONTZEPTUAK

Prestakuntza Zikloa

1DAW

Garapen inguruneak

Egileak: Jon Aguirre

2022(E)KO URRIAREN 27(A)

AURKIBIDEA

1 ¿Por qué se debe probar el software?

A un alto nivel, las pruebas de software son necesarias para detectar los errores en el software y para probar si el software cumple con los requisitos del cliente. Esto ayuda al equipo de desarrollo a corregir los errores y entregar un producto de buena calidad.

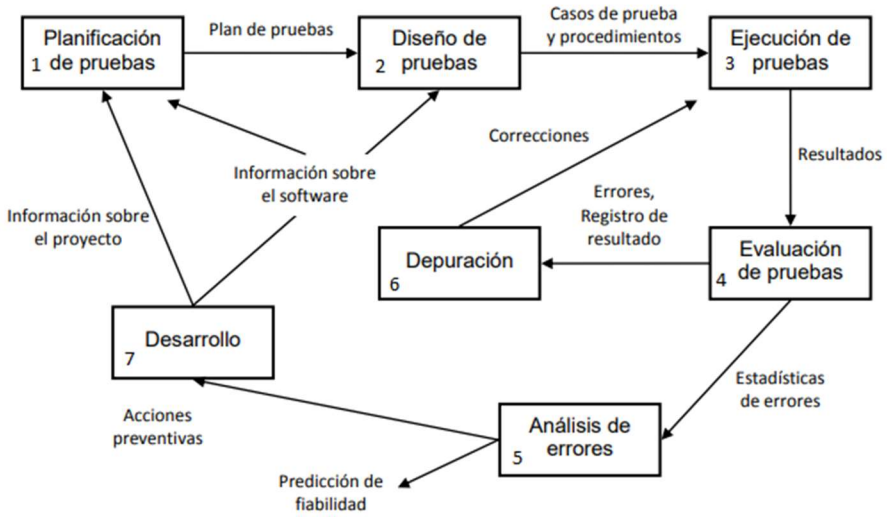
2 ¿En qué consiste la verificación del software?

La verificación de software es una disciplina de la ingeniería de software cuyo objetivo es asegurar que el software satisface por completo todos los requisitos esperados de una función específica.

3 ¿En qué consiste la validación del software?

La validación de software es un proceso que demuestra que el sistema cumple con las funciones de las cuales fue designado, de acuerdo con las especificaciones de los requisitos del usuario.

4 ¿Cuál es el proceso de la prueba de un software?



5 ¿De qué elementos consta un caso de prueba?

Los números designados en la imagen de la anterior respuesta corresponden con estos números.

1.- En primer lugar, hay que generar un plan de pruebas partiendo de la documentación del proyecto y de la documentación sobre el proyecto a probar.

2.- A partir del plan se diseñan las pruebas. Se identifican las técnicas a utilizar para probar el software.

2.5.- Generación de los casos de prueba. Se han de confeccionar los distintos casos de prueba según la técnica o técnicas identificadas previamente.

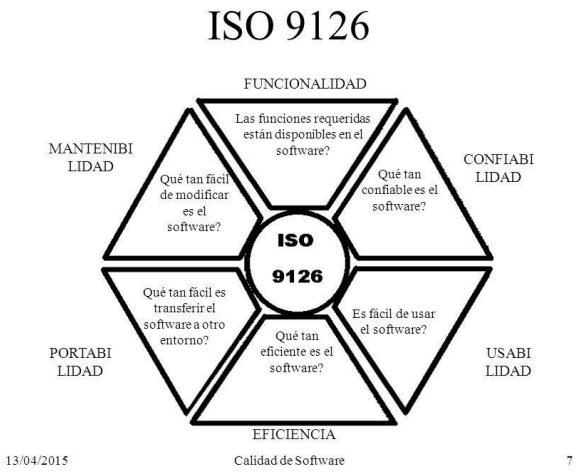
3.- Ejecución de las pruebas aplicando los casos de prueba generados previamente.

4.- Evaluación. Se identifican los posibles errores producidos al comparar los resultados obtenidos en la ejecución con los esperados. Es necesario realizar un informe con el resultado de ejecución de las pruebas, qué casos de prueba pasaron satisfactoriamente, cuáles no, y qué fallos se detectaron.

5.- El análisis de errores puede servir para predecir la fiabilidad del software y mejorar los procedimientos de desarrollo.

6.- Depuración. Trata de localizar y corregir los errores. Si se corrige un error, se debe volver a probar el software para ver que se ha resuelto el problema. Si no se consigue localizar un error puede ser necesario realizar más pruebas para obtener más información.

6 ISO 9126. Mapa conceptual



7 IEEE829

IEEE 829 Subject	Contenido
Tareas	Consiste en la división de las tareas para aplicar las pruebas al sistema
Requerimientos de ambiente	Define las herramientas necesarias para la automatización de las pruebas, hardware, software y lugar para aplicarlas.
Responsabilidades	Define a quién le pertenece cada parte del proceso de pruebas
Personal necesario y entrenamiento	Se necesitará entrenamiento de los inspectores / desarrolladores en cuanto al diseño de pruebas, herramientas, ambiente, etc.
Planificación	Fechas de inicio, hitos, completaciones, etc.
Riesgos y contingencias	¿Qué puede fallar? ¿Qué cosas fueron asumidas? ¿Qué se puede hacer si un problema mayor ocurre?
Aprobación	Determinar quién debe aprobar y revisar el plan

8 Pruebas del sistema: funcionales y no funcionales

La prueba funcional consiste en probar la 'funcionalidad' de un software o una aplicación bajo prueba.

Prueba el comportamiento del software bajo prueba. Según los requisitos del cliente, se utiliza un documento llamado especificación de software o Especificación de requisitos como guía para probar la aplicación.

Se esculpe un dato de prueba basado en él y se prepara un conjunto de casos de prueba. Luego, el software se prueba en un entorno real para verificar si el resultado real está sincronizado con el resultado esperado. Esta técnica se llama Técnica de caja negra y se lleva a cabo principalmente de forma manual y también es muy eficaz para encontrar errores.

Hay algunos aspectos que son complejos, como el rendimiento de una aplicación, etc., y la prueba no funcional comprueba la calidad del software que se va a probar. La calidad depende principalmente del tiempo, la precisión, la estabilidad, la corrección y la durabilidad de un producto en diversas circunstancias adversas.

En términos de software, cuando una aplicación funciona según las expectativas del usuario, sin problemas y de manera eficiente bajo cualquier condición, entonces se declara una aplicación confiable. Con base en estos aspectos de la calidad, es muy importante realizar pruebas bajo estos parámetros. Este tipo de prueba se denomina Técnica de caja blanca.

No es posible probar este tipo manualmente, por lo que se utilizan algunas herramientas automatizadas especiales para probarlo.

9 Pruebas unitarias

Las pruebas unitarias de software, conocidas también como unit testing o test unitarios, pueden definirse como un mecanismo de comprobación del funcionamiento de las unidades de menor tamaño de un programa o aplicación en específico.

Estas pruebas unitarias de software forman parte de la estrategia de metodología ágil del trabajo del desarrollo, donde se busca ofrecer piezas pequeñas de software en funcionamiento en un corto periodo de tiempo, con el objetivo de aumentar la satisfacción del cliente.

Es importante aclarar que este tipo de pruebas son de vital importancia para la detección de errores, ya que, sin este testeo, no podrían identificarse hasta fases más avanzadas del desarrollo, como, por ejemplo, la fase de integración. Esto implica que las pruebas unitarias de software evitan la escalada de errores en el código al identificarlas de manera temprana.

10 Pruebas de integración

Las pruebas de integración se definen como un mecanismo de testeo de software, donde se realiza un análisis de los procesos relacionados con el ensamblaje o unión de los componentes, sus comportamientos con múltiples partes del sistema (ya sea de archivos operativos) o de hardware, entre otras.

De modo que las pruebas de integración están a cargo del examen de las interfaces entre los subsistemas o los grupos de componentes del programa o aplicación que se analiza, lo que contribuye a garantizar su funcionamiento correcto.

11 Pruebas de aceptación

La prueba de aceptación o acceptance testing se encarga de establecer la oportunidad para que el cliente o negocio hago pruebas comerciales en tiempo real. Los tipos que encontramos de aceptación son:

- Alpha Testing: permite que todos los participantes del programa encuentren todos los puntos de quiebre posibles para solucionarlos antes de lanzarlo.
- Beta Testing: tiene el objetivo de que los clientes puedan probar el programa en un ambiente real.
- Operational acceptance testing (OAT): se utiliza para que los administradores del producto prueben el producto en un ambiente real.

Por otro lado, los tipos de testing de software funcionales no funcionan solos, pues también existen los no funcionales. Si deseas realizar pruebas completas sobre tu programa, no te pierdas el artículo sobre testing de software no funcionales.

12 Pruebas de regresión

Las pruebas de regresión son cualquier tipo de pruebas de software con el objeto de descubrir errores (bugs), carencias de funcionalidad, o divergencias funcionales con respecto al comportamiento esperado del software, causados por la realización de un cambio en el programa. Se evalúa el correcto funcionamiento del software desarrollado frente a evoluciones o cambios funcionales. El propósito de éstas es asegurar que los casos de prueba que ya habían sido probados y fueron exitosos permanezcan así. Se recomienda que este tipo de pruebas sean automatizadas para reducir el tiempo y esfuerzo en su ejecución.

13 Pruebas de humo

En ingeniería de software y pruebas de software, las pruebas de humo (smoke testing en inglés) son una revisión rápida de un producto de software para comprobar que funciona y no tiene

defectos evidentes que interrumpan la operación básica del mismo. Son pruebas que pretenden hacer una evaluación inicial de la calidad de un producto de software previo a una recepción formal, ya sea al equipo de pruebas (quien ejecutará una batería completa de comprobaciones) o al usuario final.

14 Pruebas de desempeño

La evaluación de desempeño es una herramienta que ayuda a mejorar la gestión de los recursos humanos dándote una mejor visión del rendimiento de cada uno de los colaboradores. Muchas veces esta evaluación es vista como algo poco importante o realizada como otro de los procesos burocráticos de la organización.

15 Pruebas de carga

Las pruebas de carga de definición generalmente se refieren a las pruebas como un subconjunto del proceso de pruebas de rendimiento del software, que normalmente también incluye varios otros tipos de pruebas, como pruebas de esfuerzo, pruebas de remojo, pruebas de picos, pruebas de resistencia, pruebas de volumen y pruebas de escalabilidad, entre otros tipos de pruebas.

Los sitios y aplicaciones de bajo rendimiento pueden tener un impacto negativo en las conversiones, transacciones y, lo que es más importante, en los ingresos. Incluso unos pocos segundos de tiempo de inactividad pueden afectar significativamente los resultados de una empresa.

16 Pruebas de estrés

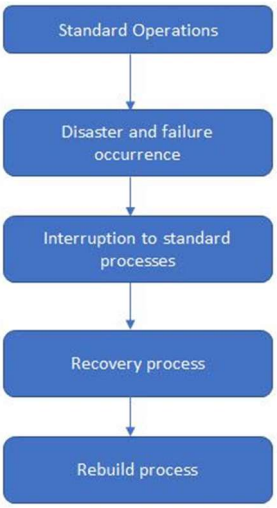
Trata de enfrentar el sistema con situaciones que demandan gran cantidad de recursos, por ejemplo, diseñando casos de prueba que requieran el máximo de memoria, incrementando la frecuencia de datos de entrada, que den problemas en un sistema operativo virtual, etc.

17 Pruebas de volumen

La prueba de volumen es un tipo de prueba de software que se realiza para probar el rendimiento o el comportamiento del sistema o la aplicación en una gran cantidad de datos. Las pruebas de volumen también se denominan pruebas de inundación y son un tipo de prueba de rendimiento.

18 Pruebas de recuperación

En este tipo de prueba se fuerza el fallo del software y se verifica que la recuperación se lleva a cabo apropiadamente.



19 Pruebas de GUI

Las pruebas de GUI, UI, o interfaz de usuario buscan validar el comportamiento de la interfaz. Es decir, las pruebas de GUI se orientan a detectar errores propios de la interfaz gráfica tales como problemas en el layout, lógica de presentación, estilos, responsividad, recursos, ortografía, etc. Las pruebas basadas en GUI buscan validar el comportamiento del sistema a través de la GUI.

20 Pruebas de configuración

La prueba de configuración es un método para probar un sistema en desarrollo en varias máquinas que tienen diferentes combinaciones o configuraciones de hardware y software. El rendimiento del sistema o una aplicación se prueba con cada una de las configuraciones de hardware y software compatibles.

Cuando decimos diferentes configuraciones de hardware y software, se atribuye a múltiples versiones del sistema operativo, navegadores, controladores compatibles, tamaños de memoria, tipos de disco duro, CPU, etc.

21 Pruebas de instalación

La prueba de los procedimientos para lograr un sistema de software instalado que se pueda utilizar se conoce como prueba de instalación. En esta instalación se incluyen pruebas de verificación de actualizaciones completas o parciales y otras características de los procesos de instalación / desinstalación. La prueba de instalación garantiza que la aplicación de software se

haya instalado correctamente con todas sus características inherentes o no. También se denomina prueba de implementación, principalmente se realiza en la fase final.

22 Pruebas de integridad de datos y BD

Pruebas de integridad de base de datos son pruebas de los métodos y procesos utilizados para acceder y gestionar datos (base de datos), para asegurar que los métodos de acceso, los procesos y las reglas de los datos funcionan como se espera y que durante el acceso a la base de datos, los datos no se corrompan, sean borrados, modificados o creados de forma inesperada.

23 Pruebas de campo

El propósito de las pruebas de campo es determinar cómo funciona una aplicación antes de lanzarla a los usuarios finales. Por lo tanto, los equipos prueban para ver cómo los usuarios finales usan la aplicación más allá del uso frecuente inicial, en un escenario del mundo real. Las pruebas se llevan a cabo usando redes móviles únicamente.

24 Pruebas funcionales

Una prueba funcional es una prueba de tipo caja negra basada en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. Las pruebas funcionales se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta el paquete informático. Dicho de otro modo, son pruebas específicas, concretas y exhaustivas para probar y validar que el software hace lo que debe y sobre todo, lo que se ha especificado.

25 Pruebas de documentación y procedimiento

En general se habla mucho de la documentación, pero no se hace, no se le asigna presupuesto, no se la mantiene y casi nunca está al día en los proyectos de desarrollo de software. Lo importante es la disponibilidad de la documentación que se necesita en el momento en que se la necesita.

Muchas veces se hace porque hay que hacerla y se escribe, con pocas ganas, largos textos, a la vez que se está convencido de estar haciendo un trabajo inútil. A veces se peca por exceso y otras por defecto. Ocurre mucho en la Web y con productos RAD. En ocasiones se olvida que el mantenimiento también debe llegar a la documentación.

https://gsitic.wordpress.com/2018/05/01/biii11-pruebas-planificacion-y-documentacion-utilizacion-de-datos-de-prueba-pruebas-de-software-hardware-procedimientos-y-datos/

26 Test limpio o positivo

Intenta mostrar que el producto satisface sus requerimientos.

27 Testo sucio o negativo

El objetivo es romper el sistema.

28 Bibliografía

https://es.slideshare.net/agrosso/software-testing-82785242

https://gsitic.wordpress.com/2018/05/01/biii11-pruebas-planificacion-y-documentacion-utilizacion-de-datos-de-prueba-pruebas-de-software-hardware-procedimientos-y-datos/

https://miso-4208-labs.gitlab.io/book/chapter4/pruebas-de-gui-o-pruebas-basadas-en-la-gui.html

https://keepcoding.io/blog/que-son-las-pruebas-unitarias-de-software/

https://keepcoding.io