Hexaware

Testing & Review

Knowledge Adquisition Document

Hugo Eduardo Jacobo Pérez

Saltillo, Coahuila 07/16/2019

Table of Contents

Agenda 1

Fundamentals of testing

Why is testing necessary?

Pienso que es necesario debido a que todos podemos cometer errores, que pueden causar cosas graves muchas veces. Por eso es necesario probar todo lo que hagamos antes de llegar a causar algo fatal.

Las personas cometemos errores todo el tiempo.

Terms of software failures

Defect: Un punto en un producto que puede no ir acorde a los requerimientos que solicita el cliente, desde defecto en el código hasta defecto físico.

Fault: Es la causa del error o defecto en un software.

Problem: Un contratiempo o punto de parada durante la ejecución de un software.

Error: Un problema que causa un resultado indeseado o fatal para el software.

Incident: Una interrupción o reducción que no estaba planeada de una aplicación o en la calidad de un producto.

Anomaly: Un resultado que resulto ser muy diferente a lo que se esperaba.

Failure: Un inconveniente de un sistema que podría provocar un error u otra consecuencia inesperada.

Inconcistency: Una forma de expresar que lo ocurrido en una tarea no tenga coherencia.

Feature: Herramientas o funcionalidades de un sistema para llevar a cabo tareas.

Bug: Un bug es una situacion que perjudica al rendimiento de un sistema en base a los requerimientos.

Un bug puede provocar:

- Pérdida de Dinero
- Pérdida de tiempo
- Perdida de reputación empresarial

Testing : Proceso de evaluar un sistema o sus componentes con la intención de determinar si cumple o no los requisitos especificados.

El objetivo de un software tester es encontrar errores, encontrarlos lo antes posible y asegurarse de que se solucionen.

Quality Process

Quality Assurance

Conjunto de actividades con el objetivo de asegurar la calidad de un software durante todas sus fases.

Quality Control

El control de calidad (QC) es un procedimiento destinado a garantizar que un producto fabricado o servicio realizado se adhiere a un conjunto definido de criterios de calidad o cumple con los requisitos del cliente o cliente.

Testing

Pruebas de un software durante su fase de desarrollo, con el fin de encontrar fallas e informar sobre ellas.

What makes a good software tester?

Requiere tener habilidades que le permitan hacer todo tipo de cosas, como escribir test plans, tener buena comunicación, etc. También saber manejar una amplia variedad de sistemas de prueba.

Exploradores: Que trate de buscar hasta el último error, colocando todo tipo de opciones.

Solucionar Problemas: Que pueda arreglar hasta los problemas mas complicados.

Creativo: Que pueda expresar diferentes maneras de pruebas, que otros no hayan pensado.

Perfeccionista: Que hasta el último detalle quede corregido.

General testing principies

- **Las pruebas revelan la presencia de bugs, no la ausencia de ellos.**

Las pruebas de software reducen la probabilidad de que queden defectos no descubiertos en el software, pero incluso si no se encuentran defectos, no es una prueba de corrección.

- **Es imposible probarlo todo.**

Se requiere realizar la cantidad correcta y especifica de pruebas basandose en una evaluacion de riesgos de la aplicacion.

Hay que ubicar cual es la operacion mas importante en la aplicacion y basarse en ella para calcular el resto.

- **Cuanto antes se empiece a probar… mejor.**

Las pruebas deben comenzar lo antes posible en el ciclo de vida del desarrollo de software. Para que cualquier defecto en los requerimientos o fase de diseño se capture en las primeras etapas. Es mucho más barato corregir un defecto en las primeras etapas de las pruebas.

- **La aglomeración de defectos.**

Esta indica que un pequeno numero de modulos contiene la mayoria de los defectos detectados, aproximadamente el 80% de los problemas se encuentran en el 20% de los módulos. Si las mismas pruebas se repiten una y otra vez, finalmente los mismos casos de prueba ya no encontrarán nuevos errores.

- **La paradoja del pesticida.**

Si se utilizan las mismas pruebas repetidas veces, el metodo utilizado sera inutil para poder probar el software y buscar nuevos defectos. Para superar eso los casos deben ser regularmente revisados y actualizados.

- **Las pruebas se deben adaptar a necesidades específicas.**

Cada una de las pruebas depende del contexto de ellas, no puedes utilizar pruebas de otro sistema o de otra funcionalidad para realizar la actual.

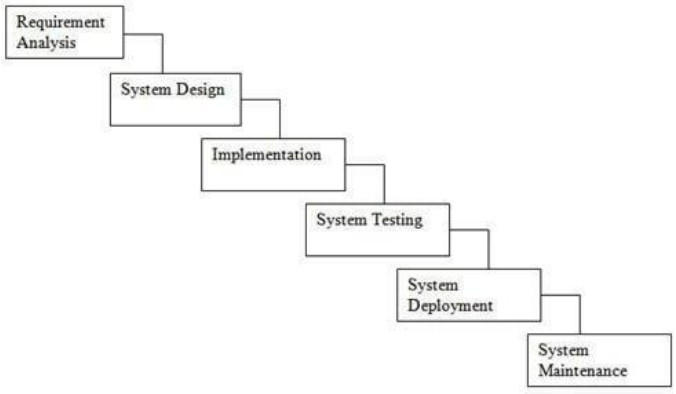
- **La ausencia de errores.**

La ausencia de errores es una falacia, encontrar y corregir defectos no ayuda si la construcción del sistema es inutilizable y no satisface las necesidades y requisitos del usuario. Por eso se debe asegurar que el software cumpla con todo lo que el cliente especifica.

Software development models

Waterfall

Es una metodología de secuencia en donde sus fases deben realizarse al momento en que una fase anterior haya terminado.



Waterfall Model - © www.SoftwareTestingHelp.com

Requirement Analysis

1. Capturar todos los requerimientos.

2. Realizar una lluvia de ideas y una manual para los requerimientos.

3. Haga la prueba de viabilidad de los requisitos para asegurarse de que los requerimientos sean verificables o no.

System Design

1. Según los requerimientos, crear el diseño.

2. Capturar los requerimientos de hardware / software.

3. Documentar los diseños.

Implementation

1. Según el diseño, crear los programas / código.

2. Integrar los códigos para la siguiente fase.

3. Unit testing en el código.

System Testing

1. Integrar el código de la unidad probada y probarlo para asegurarse de que funciona como se espera.

2. Realice todas las actividades de prueba (funcionales y no funcionales) para asegurarse de que el sistema cumple con los requerimientos.

3. En caso de cualquier anomalía, infórmalo.

4. Haga un seguimiento de su progreso en las pruebas a través de herramientas como métricas de trazabilidad, ALM

5. Informe sus actividades de prueba.

System Deployment

1. Asegúrate de que el ambiente sea el correcto.

2. Asegúrese de que no hay varios defectos abiertos.

3. Asegúrese de que se cumplan los criterios de salida de prueba.

4. Despliegue la aplicación en el entorno respectivo.

5. Realice una comprobación de integridad en el entorno una vez implementada la aplicación para asegurarse de que no se rompa.

System Maintenance

1. Asegúrese de que la aplicación esté funcionando en el entorno respectivo.

2. En caso de que el usuario se tope con un defecto, asegúrese de anotar y solucionar los problemas que se presentan.

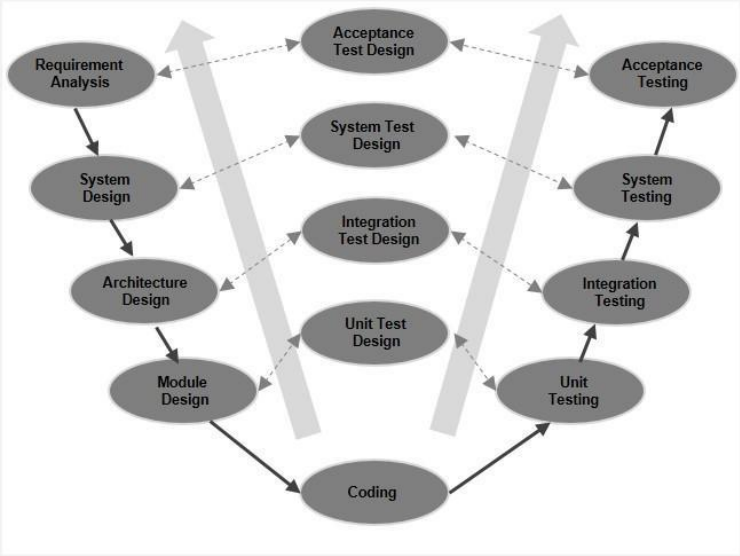
3. En caso de que cualquier problema sea arreglado; El código actualizado se implementa en el entorno.

4. La aplicación siempre se ha mejorado para incorporar más funciones, actualizar el entorno con las últimas funciones.

V-model

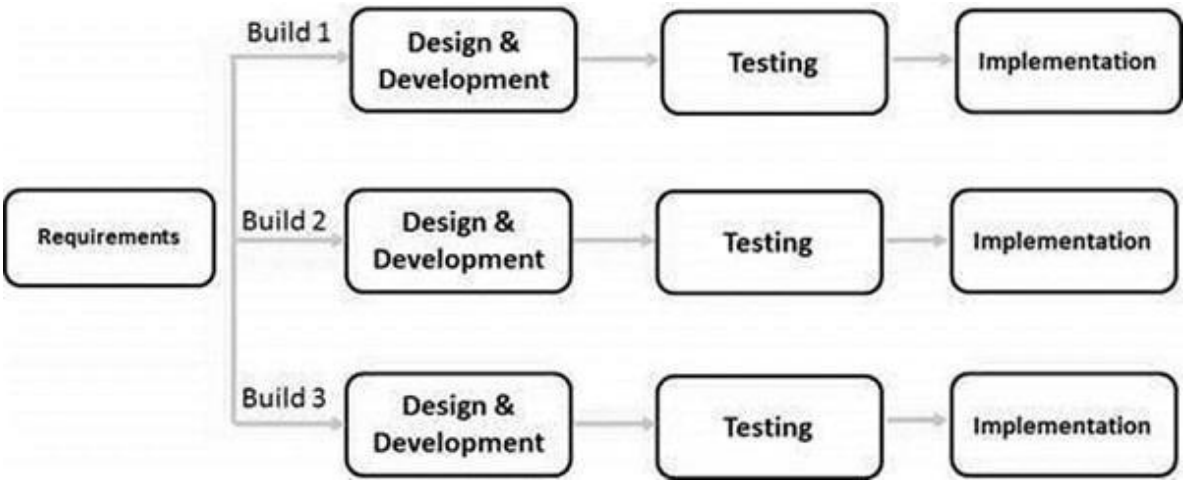
Es una metodología de secuencia en donde sus fases y actividades son presentadas en forma de V.

Esto permite que al momento en el que otra fase se lleve a cabo, también se este realizando testing de la fase anterior.



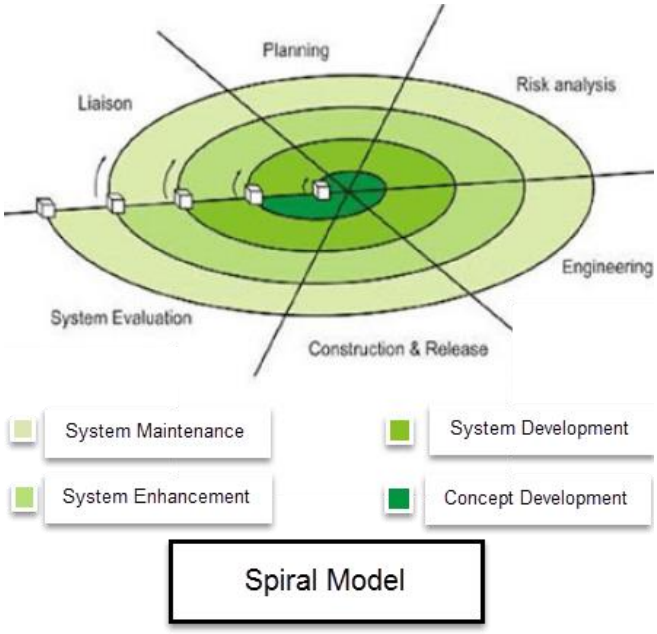
Iterative

Es una metodología que implementa pequeñas partes de un software que no necesariamente cubran todos los requerimientos, estas partes se van mejorando hasta el punto de haber colocado todos los puntos.



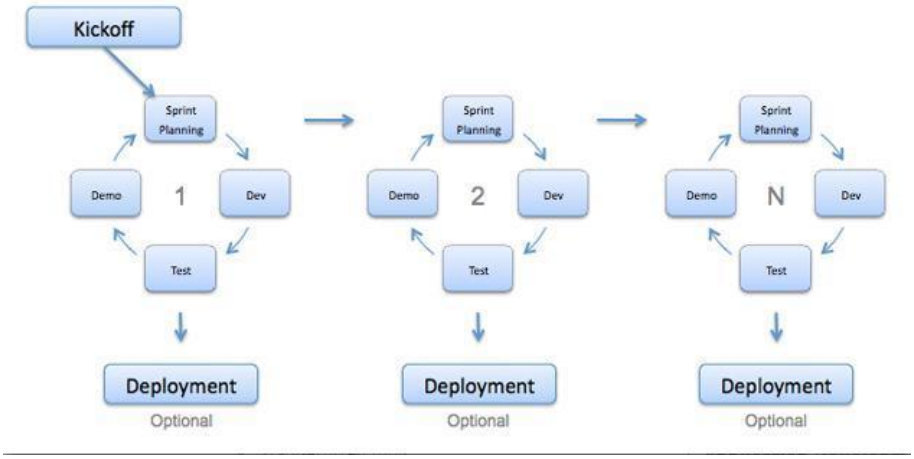
Spiral

Una metodología que realiza actividades en un ciclo constante en donde se define una meta a llevarse, al ser realizada esta tiene que ser verificada y aceptada por el cliente para poder continuar.



Agile

Es una metodología que maneja iteraciones continuas y se enfoca en los trabajos individuales y en equipo conjunto para realizar el software puesto que se realiza énfasis entre la comunicación constante entre el equipo y el cliente.



Test Levels

Unit Testing

Es la parte del testing en donde al probarse cada unidad y componentes del software, se debe verificar si funcionan tal cual se había diseñado anteriormente.

- Plan de prueba de la unidad
- Casos de prueba de la unidad
- Prueba de unidad

Se verifica todas las tareas de la unidad para encontrar defectos, daños o bugs y se realiza normalmente por los desarrolladores.

Integration Testing

Es la parte del testing en donde las unidades y componentes son combinadas y se prueban de forma conjunta, esto se realiza para poder encontrar defectos y culpables al momento de que interactúen entre sí.

- Plan de pruebas de integración
- Casos de prueba de integración / Script
- Prueba de integración

Normalmente se realiza por desarrolladores y testers individuales.

System Testing

Es el nivel en el que se prueba un software completo e integrado para poder evaluar su ejecución y que cumpla con los requerimientos acordados.

- Plan de prueba del sistema
- Casos de prueba del sistema
- Prueba del sistema

La suelen realizar los equipos de testing manual y automatizado.

Acceptance Testing

Es el nivel de prueba donde el sistema se revisa completamente para poderlo liberar pronto. El objetivo de este nivel es evaluar que se cumplan todos los requerimientos y darle una aceptación completa para poderse entregar al cliente.

- Plan de prueba de aceptación
- Casos de uso/ Checklist de aceptación
- Prueba de aceptación

La realizan miembros de la organización que desarrollan software pero que no necesariamente trabajaron en este proyecto.

Empleados que no son de la organización como clientes y testers del cliente.

Performance Testing

Es el nivel en donde se realiza el proceso de mejoras tanto de velocidad como de rendimiento del software sobre cargas de trabajo. Se realiza para diagnosticas puntos de auxilio o cuellos de botella durante las comunicaciones del sistema.

Security Testing

Son las pruebas en donde se busca encontrar las vulnerabilidades del sistema y determinar si la información que este tiene es segura ante posibles intrusos.

Hay cuatro áreas de enfoque principales que deben considerarse en las pruebas de seguridad (especialmente para sitios web / aplicaciones):

Seguridad de la red: implica buscar vulnerabilidades en la infraestructura de la red (recursos y políticas).

Seguridad del software del sistema: esto implica evaluar las debilidades en los diversos programas (sistema operativo, sistema de base de datos y otro software) de los que depende la aplicación.

Seguridad de la aplicación del lado del cliente: se trata de garantizar que el cliente (navegador o cualquier herramienta similar) no pueda ser manipulado.

Seguridad de la aplicación del lado del servidor: Esto implica asegurarse de que el código del servidor y sus tecnologías sean lo suficientemente robustos para defenderse de cualquier intrusión.

Functional testing

Verifica que todas las funcionalidades de la aplicación operen de acuerdo a como se solicito en los requerimientos, esto se realiza con pruebas de caja negra y no se revisa el código fuente de la aplicación.

Funciones de línea principal: probar las funciones principales de una aplicación

Usabilidad básica: Implica pruebas básicas de usabilidad del sistema. Comprueba si un usuario puede navegar libremente a través de las pantallas sin ninguna dificultad.

Accesibilidad: Comprueba la accesibilidad del sistema para el usuario.

Condiciones de error: Uso de técnicas de prueba para verificar condiciones de error. Comprueba si se muestran mensajes de error adecuados.

Non functional testing

Son las pruebas que revisan todos los aspectos no funcionales de una aplicación (rendimiento, facilidad de uso, fiabilidad, etc.) Está diseñado para probar la preparación de un sistema según parámetros no funcionales que nunca se abordan mediante pruebas funcionales.

Las pruebas no funcionales deben aumentar la facilidad de uso, la eficiencia, la capacidad de mantenimiento y la portabilidad del producto.

- Ayuda a reducir el riesgo de producción y el costo asociado con aspectos no funcionales del producto.
- Optimice la forma en que se instala, configura, ejecuta, administra y monitorea el producto.
- Recopile y produzca mediciones, y métricas para investigación y desarrollo internos.
- Mejorar y mejorar el conocimiento del comportamiento del producto y las tecnologías en uso.

Hexaware testing methodology

Testing life Cycle

KT & RT -> TEST SCOPING > TEST PLANNING > TEST DESIGN > TEST EXECUTION

KT : Comprenda la funcionalidad completa de la aplicación / producto a probar y demostrar al cliente que lo mismo se entiende bien para pasar a la siguiente fase.

Outputs

- Baselined Knowledge Acquisition Document
- Testing Requirements
- Test Strategy
- Acceptance Criteria
- Test plan document, test estimation, traceability matrix
- Test environment, approved test data, baselined test scenarios
- Tested product/Application, test execution reporter, test summary reports, defect log

Test Activities

Test scoping

Un alcance de prueba muestra a los equipos de pruebas de software las rutas exactas que deben cubrir al realizar sus operaciones de prueba de aplicaciones.

Un alcance de prueba bien definido puede guiarlo a lo largo del camino para entregar un buen producto de software con riesgos reducidos.

Test planning

Un plan de prueba es un documento que describe el alcance y las actividades de las pruebas de software. Es la base para probar formalmente cualquier software / producto en un proyecto.

Se requiere hacer de un análisis completo de los requerimientos para poder realizar el plan de prueba y para asegurarse de que las actividades de prueba para cada lanzamiento / ciclo de aplicación estén bien planificadas.

Los pasos de la planificación son:

- Preparar el documento del plan de pruebas.
- Revisar y repasar el plan de pruebas.
- Prueba estimada
- Preparar la matriz de trazabilidad.

Test strategy / test design

Una estrategia de prueba es un plan para definir el enfoque de prueba y responde a preguntas como qué quiere hacer y cómo lo va a lograr. También ayuda a los evaluadores a obtener una imagen clara del proyecto en cualquier momento.

El diseño es un proceso que describe como se deben hacer las pruebas. Incluye procesos para identificar los casos de prueba al enumerar los pasos de las condiciones de prueba definidas.

Las técnicas de prueba definidas en la estrategia o plan de prueba se utilizan para enumerar los pasos.

Test scripting

Un test son instrucciones (escritas usando un lenguaje de programación / scripting) que se realizan en un sistema bajo prueba para verificar que el sistema funciona como se espera. Los scripts de prueba se utilizan en las pruebas automatizadas.

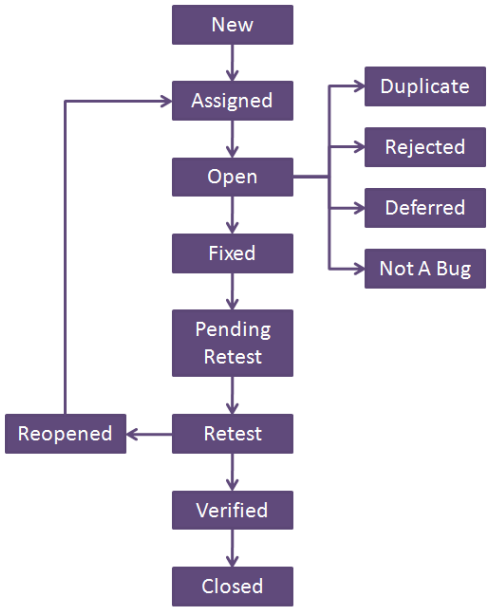
Algunos lenguajes de script utilizados en las pruebas automatizadas son:

- JavaScript
- Perl
- Python
- VbScript

Defect lifeCycle

El Ciclo de vida del defecto o el Ciclo de vida de un error es el conjunto específico de estados que un error pasa desde el descubrimiento hasta la reparación del defecto.

El ciclo de vida del defecto puede variar de una organización a otra y también de un proyecto a otro según varios factores, como la política de la organización, el modelo de desarrollo de software utilizado (como Agile, Iterative), las líneas de tiempo del proyecto, la estructura del equipo, etc.



Bug / Defect Lifecycle
http://ISTQBExamCertification.com

Defect reporting

Se documentan todas las desviaciones / defectos / casos de prueba fallidos identificados durante la prueba. Los defectos identificados por los miembros del equipo de prueba se registran en el defecto.

Defect Ratest

Los defectos corregidos son verificados, validados y cerrados por el ingeniero de pruebas correspondiente.

Defect Deferred

La solución para el defecto no se proporcionará inmediatamente.

Black Box and white box testing

Black box testing

La prueba de caja negra es un método de prueba de software en el que el tester no conoce la estructura interna / diseño / implementación del elemento que se está probando. Estas pruebas pueden ser funcionales o no funcionales, aunque generalmente son funcionales.

Este método intenta encontrar errores en las siguientes categorías:

- Funciones incorrectas o faltantes
- Errores de interfaz
- Errores en estructuras de datos o acceso a bases de datos externas.
- Comportamiento o errores de rendimiento.
- Errores de inicialización y terminación.

White box testing

Es un método de prueba de software en el que el probador conoce la estructura interna / diseño / implementación del elemento que se está probando.

El probador elige entradas para realizar recorridos a través del código y determina las salidas apropiadas.

La prueba dinámica es la prueba que se realiza cuando el sistema se está ejecutando. La prueba estática se realiza cuando el sistema no se está ejecutando.

Las pruebas de caja negra se enfocan en la funcionalidad y se realizan sin conocer el funcionamiento interno del programa.

La prueba de caja blanca busca problemas dentro del propio código.

Static and Dynamic testing

Las pruebas estáticas implican revisiones manuales o automatizadas de los documentos. Esta revisión se realiza durante una fase inicial de prueba para detectar el defecto temprano en STLC. Examina documentos de trabajo y proporciona comentarios de revisión.

Ejemplos de documentos de trabajo

- Especificaciones de requisitos
- Documento de diseño
- Código fuente
- Planes de prueba
- Casos de prueba
- Scripts de prueba
- Ayuda o documento de usuario
- Contenido de la página web

Specification terminology checklists

Palabras que nunca se deben de utilizar al momento de realizar descripciones en un caso de uso.

- Always, Every, All, None, Never.
- Certainly, Therefore, Clearly, Obviously, Evidently.
- Some, Sometimes, often, usually, ordinarilly, Customarity, most, mostly.
- Etc, And so forth, And so On, Such As.
- Good, fast, cheap, efficient, small, stable.
- Handled, Processed, Rejected, Skipped, Eliminated.

Sub boundary condition

Tipos de variables

Bit	0 or 1
Nibble	0-15
Byte	0-255
Word	0-65, 535 or 0-4 294
Kilo	1,024
Mega	1,048,576
Giga	1,073,741,524
Tera	1,099,511,627,776

Test design 2

Identifiers

Un identificador único puede ser usado para referenciar y localizar la prueba de forma más rápida.

STT_001_calcSum

Features to be tested

Una descripción de la funcionalidad de un software cubierta por el diseño de la prueba.

Approach

Una descripción del enfoque general que será usado por las funcionalidades de la prueba.

Test case identification

Una descripción de alto nivel y hacer referencias a casos de prueba específicos que serán usados para revisar las funcionalidades.

Pass/fail criteria

Describe exactamente que constituye una prueba correcta o una falla en la prueba del sistema.

Test cases 2

Identifiers

Se hace referencia a la especificación de diseño de prueba y las especificaciones del procedimiento de prueba. Que identifican un caso de prueba de forma única.

Test item

El elemento individual que se va a probar. Normalmente, hay un objeto de prueba y muchos elementos de prueba.

Input specification

Se utilizan para describir formatos de archivo descritos por el programa; renombrar los campos de archivo descritos externamente; y asigne indicadores de campo de coincidencia, interrupción de nivel y relación campo a registro.

Output specification

Las especificaciones de salida describen el registro y el formato de los campos en un archivo de salida descrito por el programa y cuándo se debe escribir el registro.

Environmental needs

Se puede considerar como todos los elementos necesarios para realizar la prueba, como las bases de datos.

Special procedural requirements

Elementos externos a la prueba que se requieren utilizar para realizarla, como plugins.

Intercase dependencies

Cuando un caso de prueba depende de que se haya realizado otra prueba antes, esa informacion se coloca en esta parte.

Test design concepts

A test condition

Eventos o funcionalidad de un sistema que se puede probar con uno o mas casos de uso.

Test Scenario

Una forma de imaginar y pensar que elementos utilizar y como para poder realizar las pruebas.

A test procedure specification

Acciones para la ejecucion de una prueba, necesarias en secuencia.

Test case design

Se requiere ser muy especifico al momento de realizar las referencias a las distintas pruebas, no basta con solo mencionarlas.

Test Coverage

Definicion de que tantos elementos cubrir y que tantos elementos probar para poder obtener una medida de la calidad del software.

Risk Analysis

El análisis de riesgos es un proceso que lo ayuda a identificar y gestionar problemas potenciales que podrían socavar iniciativas o proyectos empresariales clave.

Para realizar un Análisis de Riesgos, primero debe identificar las posibles amenazas a las que se enfrenta y luego estimar la probabilidad de que estas amenazas se materialicen.

Bugs lifecycle

Bugs

Ocurren cuando uno o más de las siguientes reglas son verdaderas.

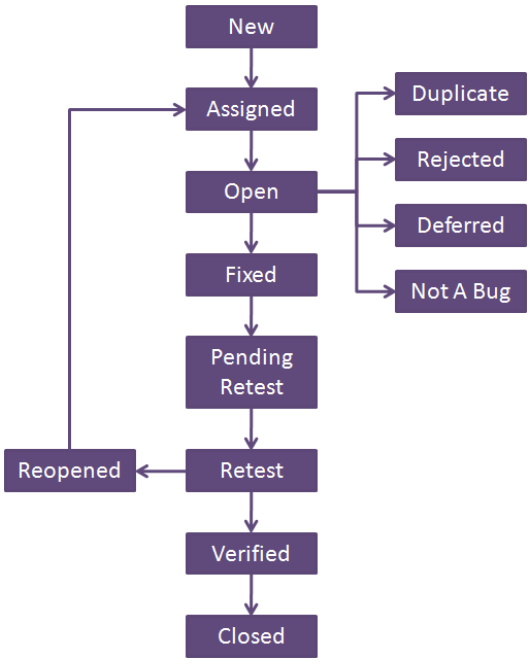
1. El software no hace algo que la especificación dice que debe hacer.
2. El software hace algo que la especificación del producto dice que no debe hacer.
3. El software hace algo que la especificación no menciona.
4. El software no hace algo que la especificación no menciona pero debería.
5. El software es difícil de entender, usar por el tester.

Siempre se deben de reportar los bugs tan pronto como sea posible.

El ciclo de vida del defecto es un ciclo por el que pasa un defecto durante su vida útil.

Se inicia cuando se encuentra el defecto y termina cuando se cierra un defecto, después de asegurarse de que no se reproduzca. El ciclo de vida del defecto está relacionado con el error encontrado durante la prueba.

El ciclo de vida del defecto puede variar de una organización a otra y también de un proyecto a otro según varios factores, como la política de la organización, el modelo de desarrollo de software utilizado (como Agile, Iterative), las líneas de tiempo del proyecto, la estructura del equipo, etc.



Bug / Defect Lifecycle
http://ISTQBExamCertification.com

Glosary

Assurance

Las pruebas de control de calidad se abrevian como pruebas de control de calidad.

La garantía de calidad se conoce popularmente como Pruebas de control de calidad, es una actividad para garantizar que una organización esté proporcionando el mejor producto o servicio posible a los clientes. El control de calidad se enfoca en mejorar los procesos para entregar productos de calidad al cliente.

Troubleshooters

La solución de problemas de software es el proceso de escanear, identificar, diagnosticar y resolver problemas, errores y errores en el software.

Es un proceso sistemático que apunta a filtrar y resolver problemas, y restaurar el software a su funcionamiento normal. Es una subcategoría de solución de problemas de TI.

Persuasive

Personas que tienen la confianza suficiente para sentirse cómodos en su propia piel. Al concentrarse en lo que lo impulsa y lo hace feliz como persona, se convierte en una persona mucho más interesante y persuasiva que si trata de ganarse a la gente intentando ser la persona que desea que sea.

Paradox

La paradoja del pesticida es cuando una pequeña cantidad de módulos contiene la mayoría de los errores detectados o muestra la mayoría de los fallos operativos. Paradoja del pesticida: si las mismas pruebas se repiten una y otra vez, eventualmente las mismas pruebas ya no encontrarán nuevos errores.

Model

Las estrategias y los tipos de prueba utilizados para certificar que la Aplicación bajo prueba cumple con las expectativas del cliente. Las metodologías de prueba incluyen pruebas funcionales y no funcionales para validar el AUT.

Modules

Un módulo es parte de un programa. Los programas se componen de uno o más módulos desarrollados independientemente que no se combinan hasta que el programa está vinculado. Un solo módulo puede contener una o varias rutinas.

Scoping

Un alcance de prueba muestra a los equipos de pruebas de software las rutas exactas que deben cubrir al realizar sus operaciones de prueba de aplicaciones.

Environment

Un entorno de prueba es una configuración de software y hardware para que los equipos de prueba ejecuten los casos de prueba. En otras palabras, es compatible con la ejecución de pruebas con hardware, software y redes configuradas.

El entorno de prueba se configura según la necesidad de la Aplicación bajo prueba.

Retest

Defect Retest es una parte del ciclo de defectos y bugs cuando los defectos corregidos son verificados, validados y cerrados por el ingeniero de pruebas correspondiente.

Deferred

Defect Deferred es una parte del ciclo de defectos y bugs que indica que la solución para el defecto no se proporcionará inmediatamente.