Los requerimientos de confiabilidad y seguridad pueden considerarse como requerimientos de protección

Requerimientos de confiabilidad y seguridad:

- Requerimientos funcionales
- Requerimientos no funcionales

Requerimientos funcionales: Se refiere a las funcionalidades que deben de ser implementadas para garantizar que el sistema sea confiable y seguro.

Requerimientos no funcionales: Describe las características que un software y su entorno de ejecución debe de tener para garantizar confiabilidad y seguridad.

Ejemplo:

- Disponibilidad del sistema
- Tolerancia a los fallos
- Capacidad de recuperación

Requerimientos 'no debe':

- No debe de modificar permisos (Roles del sistema)

NOTA: Acá se debe de evaluar los niveles de seguridad de un sistema. Esto para garantizar la confiabilidad e integridad de datos y sus procesamientos.

Requerimientos dirigidos por riesgos
(Identificación del riesgo , análisis del riegos, descomposición del riesgo , reducción del riesgo)

1- Identificación: Son los riesgos potenciales al sistema. Éstos dependen del entorno en que se usa el sistema. Pueden surgir riesgos de interacciones entre el sistema y las condiciones extrañas de su entorno operacional.

2. Análisis y clasificación: Cada riesgo se considera por separado. Los riesgos pueden eliminarse porque es improbable que surjan o porque no se pueden detectar con el software

3. Descomposición: se enfoca en el descubrimiento de los eventos que quizá conduzcan a que ocurra un peligro.

4. Reducción: se basa en el resultado del análisis del peligro y lleva a la identificación de requerimientos de protección.

Para sistemas grandes, el análisis de riesgo puede estructurarse en fases donde cada fase considera diferentes tipos de riesgos:

1. Análisis de riesgo: Implica la identificación de riesgos que pueden derivar en accidentes o incidentes. Entonces se generan requerimientos de sistema para garantizar que dichos riesgos no ocurran o que, si ocurren, no conduzcan a un incidente o accidente

2. Análisis de riesgo de ciclo de vida: En el desarrollo y principalmente a los riesgos surgidos por decisiones de diseño del sistema.

3. Análisis de riesgo operativo:. Se preocupa por la interfaz de usuario del sistema y los riesgos de error del operador.

Las passphrases se consideran más seguras que las passwords. Son más difíciles de descifrar por un atacante o de descubrir a través de un sistema automatizado de rompimiento de contraseñas.

Identificación de peligro. Se trata al considerar los diferentes tipos de peligros. como los físicos, eléctricos, biológicos, de radiación, de falla de servicio.

Enfoque dirigido por peligros. para comprender los requerimientos de seguridad de un sistema. Se identifican los peligros potenciales y se desglosan

Valoración del peligro se enfoca en comprender la probabilidad de que exista un peligro y las consecuencias de que ocurriera un accidente asociado con dicho peligro.

1. **Los riesgos intolerables:** son aquellos que amenazan la vida humana.

2. **Los riesgos ALARP:** Tienen consecuencias menos serias, aun cuando son graves, tienen una muy baja probabilidad de sucedan.

3. **Los riesgos aceptables:** Los accidentes asociados derivan por lo general en un daño menor.

La fiabilidad: Es diferente de la seguridad y la protección, pues se considera un atributo mensurable del sistema. Esto es, se puede especificar y comprobar si se logró la fiabilidad requerida.

Métricas de fiabilidad. puede definirse como una probabilidad de que una falla del sistema ocurrirá cuando un sistema está en uso dentro de un entorno operacional especificado.

Probabilidad de falla a pedido (POFOD). Probabilidad de que la demanda por un servicio de un sistema derive en una falla del sistema. De este modo, POFOD = 0.001

Tasa de ocurrencia de fallas (ROCOF). Establece el número probable de fallas de sistema que se observan en relación con cierto tiempo.

AVAIL: Es la probabilidad de que un sistema esté en operación cuando se haga una demanda por servicio.

Requerimientos de fiabilidad no funcionales: Especificaciones cuantitativas de la fiabilidad y disponibilidad requeridas de un sistema, calculadas mediante métricas.

Especificación de fiabilidad funciona: La especificación de fiabilidad funcional incluye identificar los requerimientos que definen las restricciones y las características que contribuyen a la fiabilidad del sistema.

Requerimientos de fiabilidad funcional:

1- **De comprobación:** Garantiza que las entradas incorrectas o fuera de rango se detecten antes de que las procese el sistema.

2- **De recuperación:** Requerimientos que se implementan para ayudar al sistema a recuperarse luego de que ocurre una falla. Ej: copias del sistema.

3- **De redundancia:** Características del sistema que aseguran que la falla en un solo componente no conduzca a una pérdida completa del servicio.

Tipos de requerimientos de seguridad:

1. Los requerimientos de identificación especifican si un sistema debe o no debe identificar a sus usuarios antes de interactuar con ellos.

2.Identificación: Explican cómo se identifica a los usuarios.

3. Autorización: Detallan los privilegios y permisos de acceso de los usuarios identificados.

4. Inmunidad: definen cómo un sistema debe protegerse a sí mismo contra amenazas.

5. De integridad: Describen cómo puede evitarse la corrupción de datos.

La política de seguridad de la organización es fundamental para la valoración y gestión del riesgo, ya que establece lo que se permite y no se permite en todos los sistemas para identificar amenazas potenciales. Por otro lado, las especificaciones formales son modelos matemáticos que describen los requerimientos del sistema y se usan en métodos formales de desarrollo de software para garantizar que el programa resultante cumpla con la especificación. Las especificaciones formales son costosas, por lo que se enfocan en componentes críticos del sistema y tienen ventajas como una comprensión profunda y pormenorizada de los requerimientos, la capacidad de analizarse automáticamente para encontrar inconsistencias, y la reducción de costos de pruebas.

Ventajas.

1- Comprensión profunda y pormenorizada de los requerimientos del sistema.

2- puede analizarse de manera automática para descubrir inconsistencias y aquello que no se completó.

3- se garantiza que el programa resultante cumpla su especificación.

4- Los costos de las pruebas suelen reducirse porque el programa se verificó contra su especificación.

MEDICIÓN Y MÉTRICAS DEL SOFTWARE

Descripción: Medir las métricas del software es como hacer una especie de examen para ver si el software es bueno. Se usan números para ver cosas como cuán grande, complicado o rápido es el software. Esto nos ayuda a entender si el software es de buena calidad y a detectar problemas. Las métricas también nos ayudan a tomar decisiones informadas en la gestión de proyectos de desarrollo de software, como decidir qué diseño o implementación es mejor o hacer seguimiento del progreso del proyecto.

Es importante seleccionar con cuidado las métricas del software y asegurarse de que estén alineadas con los objetivos del proyecto y las necesidades de la organización. Las métricas del software son una herramienta útil, pero no son la única medida de calidad del software. Es importante utilizar una combinación de prácticas para gestionar la calidad del software de manera integral. Cada proyecto de software es diferente, por lo que no hay una única métrica que se aplique a todos. Es crucial elegir y utilizar las métricas adecuadas para cada proyecto de software específico.

EJEMPLOS DE MÉTRICAS DEL SOFTWARE

1. Cantidad lineas de código
2. Complejidad ciclomatica
3. Cantidad de defectos encontrados y corregidos
4. Tasa de cumplimiento de requisitos
5. Eficiencia en uso de recursos
6. Usabilidad
7. Manteniabilidad
8. Confiabilidad del software

ALGUNOS BENEFICIOS POTENCIALES DE LA MEDICIÓN DE MÉTRICAS DEL SOFTWARE INCLUYEN:

• **Control y seguimiento del proyecto:** las métricas del software permiten a los equipos de desarrollo y gestión identificar posibles problemas y tomar medidas oportunas para corregirlos.

• **Identificación temprana de defectos:** las métricas del software ayudan a identificar áreas problemáticas en el software y tomar medidas pro-activas para abordarlas antes de que se conviertan en defectos costosos en la etapa de producción.

• **Toma de decisiones informadas:** las métricas del software proporcionan datos cuantitativos para tomar decisiones informadas en la planificación, diseño, desarrollo, asignación de recursos y priorización de actividades.

• **Mejora continua:** el seguimiento y análisis de las métricas del software a lo largo del tiempo identifican áreas de mejora y facilitan la adopción de prácticas de mejora continua en el desarrollo del software.

MÉTRICAS ESTÁTICAS DE PRODUCTOS DE SOFTWARE.

Fan-in/Fan-out: son medidas que indican cuántas funciones o métodos llaman a otra función o método (Fan-in) y cuántas funciones o métodos son llamados por una función o método (Fan-out). Una alta cantidad de Fan-in indica que hay un acoplamiento estrecho y cambios extensos, mientras que un alto valor de Fan-out sugiere complejidad en la lógica de control.

Longitud de código: es una medida del tamaño de un programa, mientras más grande sea el código, más complejo y propenso a errores será el componente. Esta métrica se utiliza para predecir la probabilidad de que haya errores en el componente.

Complejidad ciclomática: mide la complejidad del control de un programa, lo que puede afectar la comprensibilidad del mismo.

Longitud de identificadores: mide el promedio de caracteres en los nombres de variables, clases, métodos, etc. Cuanto más largos sean, más significativos y comprensibles serán.

Profundidad de anidado: mide la cantidad de enunciados "if" anidados en un programa, y los enunciados "if" profundamente anidados son más propensos a errores y más difíciles de entender.

Índice Fog: es una medida de la longitud promedio de las palabras y oraciones en los documentos. Cuanto mayor sea el valor del índice Fog, mayor será la dificultad de comprensión del documento.

QUE ES SQA

El equipo de SQA garantiza que el software cumpla con los estándares de calidad y sea apto para su uso en producción.

El equipo de SQA (Aseguramiento de Calidad de Software) es responsable de garantizar que el software cumpla con los requisitos de calidad y funcione correctamente. Para lograrlo, realizan pruebas y verificaciones en todas las fases del ciclo de vida del software, desde la planificación hasta la implementación. También se aseguran de que se sigan los procesos y estándares adecuados durante el desarrollo del software, y trabajan para identificar y resolver problemas antes de que lleguen a los usuarios finales. En resumen, el equipo de SQA trabaja para garantizar que el software sea de alta calidad, funcione correctamente y cumpla con las expectativas de los usuarios.

ACTIVIDADES DE SQA

• Establecimiento de procesos y metodologías de pruebas.

• Diseño, implementación y ejecución de pruebas manuales y automatizadas.

• Realización de pruebas de regresión.

• Establecimiento de métricas y estándares de calidad.

• Identificación y documentación de problemas y defectos para su corrección.

ROLES DE SQA

• Líder de SQA: lidera el equipo y coordina las actividades de aseguramiento de calidad.

• Ingeniero de Pruebas: diseña, implementa y ejecuta pruebas manuales y automatizadas.

• Especialista en Automatización: desarrolla y mantiene pruebas automatizadas.

• Analista de Calidad: mide la calidad del software a través de métricas y estándares.

• Gerente de Calidad: asegura que la calidad del software cumpla con los estándares de la organización.

RESPONSABILIDADES DE SQA

• Asegurar que el software cumpla con los estándares de calidad definidos.

• Identificar y documentar problemas y defectos en el software.

• Colaborar con otros miembros del equipo de desarrollo para mejorar la calidad del software.

• Garantizar la eficiencia y eficacia de las pruebas y documentar los resultados adecuadamente.

• Proporcionar informes regulares sobre la calidad del software y las pruebas realizadas.

• Verificar y validar requisitos, planificar pruebas, evaluar herramientas de prueba, gestionar configuración y realizar auditorías de calidad.

RESUMEN

El equipo de SQA es responsable de garantizar que el software cumpla con los estándares de calidad y requisitos del usuario final, mediante la realización de pruebas eficientes y eficaces, la identificación y documentación de problemas y defectos en el software, y la colaboración con el equipo de desarrollo para resolver problemas y mejorar la calidad del software.