PRINCIPIOS DE PREVENCIÓN DE DEFECTOS

Emanuel Alejandro Gutierrez Romero

02/09/23

COMPUTACION TOLERANTE A FALLAS

Prof. Michel Emanuel Lopez Franco

Introducción

Si bien es cierto que debemos saber como manejar los errores, está en nosotros también como desarrolladores el prevenirlos, por que se sabe que es mejor prevenir que lamentar. Es por ello por lo que presento aquí algunas formas de prevenir fallos.

Desarrollo

Para prevenir antes que lamentar tenemos presentes los siguientes puntos.

- Involucrar a los usuarios finales lo antes posible: Varios estudios han encontrado que la participación del usuario final es clave para la estabilidad de los requisitos y el éxito del proyecto del software.

- Crear un prototipo desechable: Crear un prototipo de interfaz de usuario desechable; presentar el prototipo a usuarios finales reales. Obtener comentarios. Revisar el prototipo hasta que el usuario esté entusiasmado con el sistema. Luego construir el sistema. Este enfoque se correlaciona con una buena estabilidad de los requisitos y un bajo coste del sistema.

- Entregar el software de forma incremental en lugar de hacerlo todo a la vez: Escribir el código de producción para una pequeña parte del sistema. Poner esta funcionalidad frente al usuario. Revisar los requisitos, el diseño y el código hasta que el usuario esté entusiasmado con el sistema. Este enfoque no elimina por completo la dinámica de defecto-aumento de costos, para acortar el ciclo de retroalimentación desde los requisitos hasta la retroalimentación del usuario de una manera que reduce la cantidad de dependencias posteriores que se basarán en un trabajo erróneo.

- Realizar un taller de requisitos: Se ha descubierto que las técnicas rápidas de obtención de requisitos, como las sesiones JAD, son una forma eficaz de acortar el tiempo necesario para recopilar requisitos precisos y, al mismo tiempo, reducir la volatilidad de los requisitos en sentido descendente.

- Realizar análisis de casos de uso: En lugar de estar satisfecho con la primera explicación de los usuarios sobre lo que quieren que haga un sistema, examinar los patrones de usos esperados del sistema para comprender mejor las necesidades reales del usuario.

- Primero crear el manual de usuario: La mayoría de los usuarios suelen entender mejor la explicación proveniente de un manual de usuarios a una especificación de requisitos tradicional

Para implantar un proceso de pruebas basado en la prevención de defectos es necesario definir qué parámetros van a servir para decidir si un defecto debe ser

analizado o no. Partiendo de la base de que los recursos son limitados, no es viable analizar la causa raíz de todos los defectos, sino sólo de aquellos que por su tipología merecen ese esfuerzo. Los parámetros que habitualmente se utilizan son: el daño potencial del defecto (económico o de imagen), frecuencia de ocurrencia, el coste que supone solucionarlo y el impacto del defecto en el conjunto de la aplicación.

Por otra parte, es necesario disponer de técnicas, mecanismos, herramientas y formación para que el análisis de la causa raíz de los defectos pueda llevarse a cabo de forma sistemática y formalizada. Diagramas de Ishikawa, diagramas de causa-efecto, esquemas de clasificación de defectos o la técnica de los 5 porqués, son algunos ejemplos de técnicas que permiten analizar cuál es la causa raíz de un defecto.

Con los parámetros y mecanismos de prevención implantados se estará en disposición de incorporar esta práctica al proceso de pruebas. Lo habitual es que al final de cada uno de los niveles de pruebas definidos en la organización se seleccionen, en base a los parámetros establecidos, los defectos a analizar. Para su análisis se aplicarán las técnicas definidas hasta llegar a identificar la causa raíz que ha provocado el defecto. Una vez conocida dicha causa, el siguiente paso será definir una o varias acciones correctivas que puedan evitar que un defecto similar ocurra. Como es lógico, las acciones correctivas se focalizan en las etapas anteriores al punto del ciclo de vida en la que se detectó el defecto, ya sea fase de requisitos, análisis, diseño o construcción.

Conclusiones

Puede ser que este trabajo se vea mas ambientado a la ingeniería en software, pero es algo realmente interesante el saber que todo lleva una serie de procesos y saber que mas que hacer simples scripts para nuestra materia de programación, ahora tratamos con software completo y que el usuario puede o no estar relacionado con el tema y puede o no saber como se solucionan problemas o saber si quiera si están ocurriendo.

Bibliografía

- Mtp. (2023). Prevención de defectos en el aseguramiento de la calidad software. *MTP*. https://www.mtp.es/blog/testing-software/prevencion-de-defectos-en-el-aseguramiento-de-la-calidad-del-software/
- GeeksforGeeks. (2020). Defect prevention methods and techniques. *GeeksforGeeks*. https://www.geeksforgeeks.org/defect-prevention-methods-and-techniques/