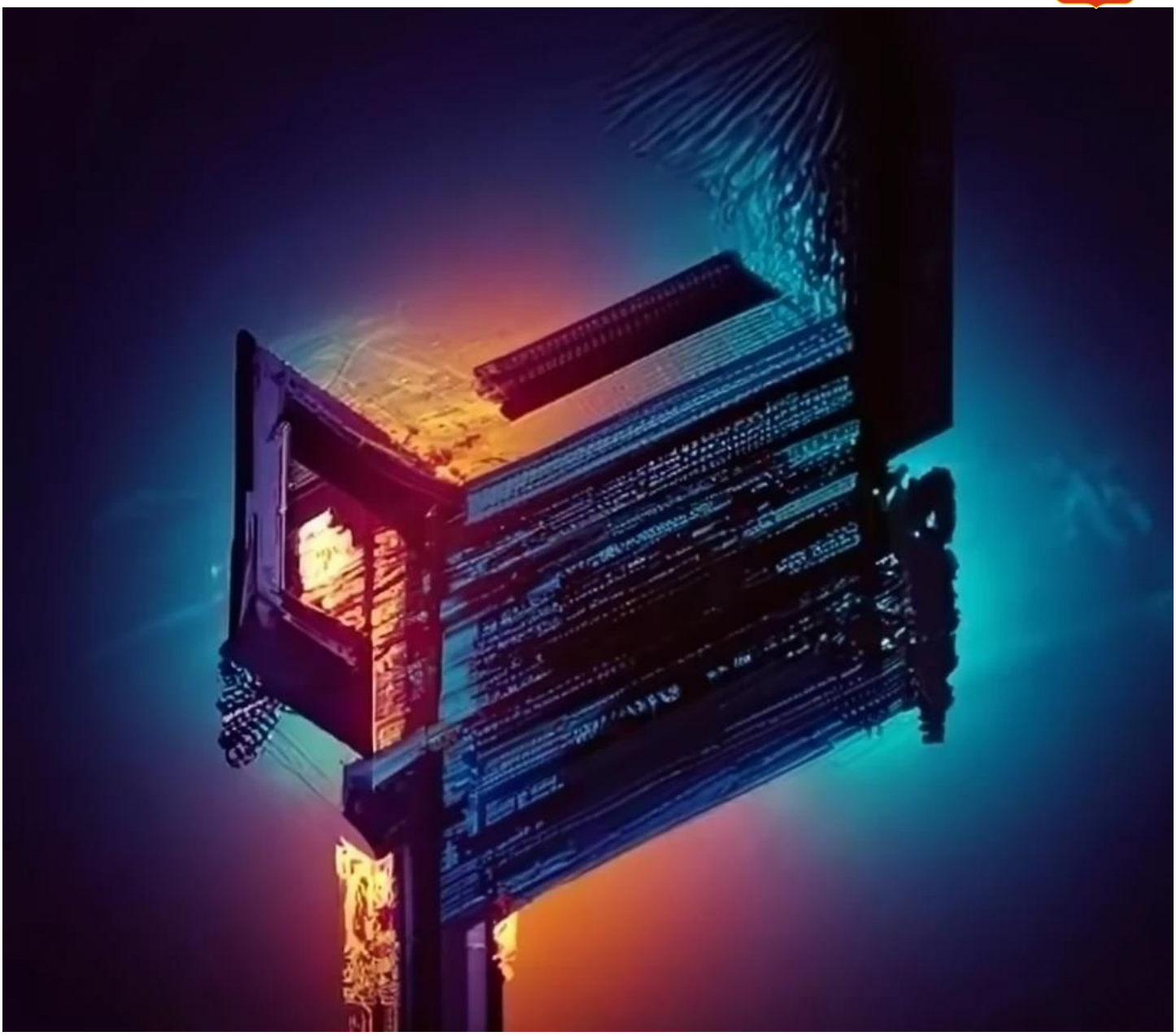
Universidad de Guadalajara
Centro Universitario de Ciencias Exactas e Ingenierías



Sección: D06 Profesor: MICHEL EMANUEL LOPEZ FRANCO
Tema: Principios de prevención de defectos
Ciclo: 2024A

Introducción

Dentro de la computación y la informática para la investigación de la tolerancia de fallas en software es importante saber de prevención de defectos por los cuales nos podemos evitar muchos problemas para lo cual tenemos que estar alerta en los cambios y la administración de la programación tomando en cuenta si se trabaja solo o con un equipo para la elaboración de un proyecto.

El desafío clave para cualquier organización de software es desarrollar un producto de software.
con menos defectos posteriores a la implementación. Además, si los defectos llegan hasta el despliegue entonces el proyecto correrá un mayor riesgo en términos de costos y tiempos. A
Una pequeña cantidad de esfuerzo inicial en la calidad del software definitivamente ahorrará una buena cantidad de
El costo y el tiempo se comparan con la estrategia de reconocimiento y eliminación de defectos.

Contenido

La forma más eficaz de gestionar los defectos es para impedir su introducción inicial. En la PSP, Hay tres diferentes pero que se apoyan mutuamente.
formas de prevenir defectos. La primera es tener datos de registro del ingeniero sobre cada defecto que encuentren y arreglar. Luego revisan estos datos para determinar qué causó los defectos y cómo realizar el proceso cambios para eliminar estas causas. Midiendo sus defectos, los ingenieros son más conscientes de sus errores, son más sensibles a consecuencias y tienen los datos necesarios para Evitar cometer los mismos errores en el futuro.
La rápida disminución inicial del total de defectos durante el Los primeros programas de cursos de PSP indican la efectividad de este método de prevención.

Dentro de lo que se puede considerar la composición de lo que todo desarrollador debe tener en cuenta para la creación de proyectos en los cuales se deba ser tolerante a fallas se toman los siguientes puntos explicados de manera corta para tomar como pasos dentro de lo que se necesite en cada proyecto distinto.

Diseño Robusto: Un diseño sólido y bien pensado puede prevenir muchos errores y problemas antes de que ocurran. Esto implica pensar en casos extremos, manejo de errores, y diseñar con la escalabilidad y la seguridad en mente.

Pruebas de Software: Las pruebas exhaustivas del software, incluyendo pruebas unitarias, pruebas de integración, pruebas de regresión y pruebas de estrés, pueden ayudar a identificar y corregir errores antes de que lleguen a producción.

Revisión de Código: La revisión de código por pares o en equipo puede ayudar a identificar problemas de lógica, errores de sintaxis y otros problemas antes de que el código se implemente en producción.

Automatización de Procesos: La automatización de tareas repetitivas y propensas a errores puede reducir la probabilidad de fallos humanos y mejorar la consistencia del sistema.

Control de Cambios: Un proceso formalizado de control de cambios puede ayudar a gestionar y documentar los cambios realizados en el sistema, lo que reduce la probabilidad de introducir errores.

Diseño a Prueba de Fallos: Utilizar técnicas de diseño que minimicen la probabilidad de fallos y maximicen la capacidad del sistema para tolerar fallos, como la redundancia, la diversidad y la detección y recuperación de errores.

Monitoreo y Supervisión: Implementar sistemas de monitoreo y supervisión que alerten sobre problemas potenciales en tiempo real, lo que permite una respuesta rápida ante fallos y problemas.

Referencias

- (S/f). Ajol.info. Recuperado el 6 de febrero de 2024, de https://www.ajol.info/index.php/star/article/view/98848/88107