Computación Tolerante a Fallas

Sección D06

Orthogonal Defect Classification (ODC)

Fernández Venegas David Guadalupe

Código: 216437352

Profesor: López Franco Michel Emanuel

Introducción:

El Orthogonal Defect Classification (ODC) es un enfoque estructurado para clasificar y analizar defectos en el desarrollo de software. Surgió a finales de la década de 1980 y principios de la década de 1990 por IBM. Originalmente, fue desarrollado para mejorar la calidad del software en los sistemas de conmutación telefónica. Desde entonces, ha sido adoptado y adaptado por varias organizaciones de desarrollo de software como una herramienta efectiva para comprender y mejorar el proceso de desarrollo y aseguramiento de la calidad.

Desarrollo:

Orthogonal Defect Classification (ODC) es un sistema de medición multidimensional con características tanto cualitativas como cuantitativas. Y actualmente es muy utilizado en la industria del software. ODC es independiente del modelo de proceso, del lenguaje y del dominio. Varias corporaciones han informado sobre aplicaciones de ODC en una variedad de plataformas y procesos de desarrollo, que van desde procesos de desarrollo en cascada, en espiral, cerrados y ágiles.

ODC puede ser usada para mejorar el círculo de calidad del proceso de prevención de defectos, lo cual se ha demostrado que resulta en un alto grado de ahorro en costos de análisis, ya que el proceso de clasificación permite encontrar de forma retrospectiva la causa y efecto en un muy corto tiempo en lo que se refiere al tiempo que toma a un grupo de analistas efectuar un detallado análisis de la causa raíz del defecto. Además, proporciona un sistema para convertir esa información en métricas que proporcionen estadísticas que permitan definir de manera más acertada los puntos que necesitan ser tratados.

El ODC se basa en el principio de que los defectos en el software pueden clasificarse en categorías independientes entre sí, o "ortogonales". Estas categorías son mutuamente excluyentes y exhaustivas, lo que significa que un defecto específico puede clasificarse en una sola categoría y que todas las posibles categorías están cubiertas. Las clases de defectos son creadas en función de las especificaciones del proyecto, el objetivo es que dichas clases puedan explicar el progreso del producto a través del proceso y que puedan ser utilizadas a lo largo de ciclo de desarrollo del proyecto.

Trabajando con el ODC:

1. Declarar los tipos de defectos

Los tipos de defectos deben definir o deben capturar, en su nombre, el significado de lo que se arregló. Una vez definidos los tipos de defecto deben analizarse y para ello se debe mapear el tipo de defecto con el proceso, independientemente del estado en el que se encuentre y del producto que se utiliza, lo importante es encontrar la relación entre los tipos de defecto y la etapa del proceso en la que se podría detectar. Los pasos para definir una clase son:

- Definir las métricas necesarias
- Validar los puntos a corregir tomando en consideración la experiencia en otros proyectos
- Mejorar el sistema de mediciones cuando se aprenden nuevos métodos a través de los experimentos

2. Definir atributos

Una vez definidas las clases, deben determinarse los atributos o condiciones que se usarán para definir los defectos puntuales, deben ser adecuados para poder categorizar cada uno de ellos de manera correcta y mapearlos a la etapa en la que pudo haberse prevenido el error. Existen dos tipos de atributos que se recomiendan para la definición de errores:

Tipo de defecto: Se declaran los tipos donde un programador pueda categorizar un defecto de manera sencilla y sin confusiones. Se recomienda usar los siguientes tipos de defecto:

- Código faltante: Se refiere a que la funcionalidad no fue implementada, por lo que la verificación encontró el defecto.
- Código incorrecto: Hace referencia cuando el código fue implementado pero tiene errores que llevaron a encontrar el defecto.
- Errores de función: Se refiere a problemas de capacidad, configuración o comunicación entre capas de la arquitectura.
- Asignación: Pocas líneas de código, como inicialización de variables.
- Interface: Errores con la interfaz final del usuario o interacción con otros componentes.
- Validación: Lógica del sistema que falló en validar configuración errónea.
- Sincronización: Manejo de recursos compartidos.
- Build, empaquetado y merge: Errores con librerías externas utilizadas, manejo de cambios o control de versiones.

Tipo de disparadores: Son condiciones que saca a la superficie un defecto. Idealmente, la distribución de los disparadores para los defectos de campo debe ser similar al encontrado en las pruebas del sistema. Una discrepancia puede indicar deficiencias en el ambiente de pruebas.

3. Clasificar causas y efectos

Es importante recolectar la información no solo de los disparadores, sino también de todos aquellos inconvenientes derivados del error que afectan directamente al desempeño del aplicativo.

Causa: Atributo de clasificación ortogonal que describe un defecto para:

- El proceso de desarrollo
- El proceso de verificación

Efecto: Atributos o métricas que están afectando al producto o a los procesos, por ejemplo:

- Áreas de impacto
- Densidad de defectos
- Rehacer arreglos de código

4. Planteamiento general del método

En el contexto de la metodología, un defecto se refiere a un cambio necesario en el software, básicamente categoriza los defectos en clases, y señala que parte del proceso requiere atención. Al corregir un defecto, el programador le asigna un "tipo", que es trazado a una o varias fases del proceso de desarrollo. Añadir una nueva característica es distinto a cambiar un par de líneas de código para corregir el valor de una variable. Al ser la clasificación un proceso manual, se busca que el conjunto de tipos de defectos sea ortogonal (que no se junten entre ellos) para minimizar los errores y evitar confusiones. Además, los tipos de defectos deben ser generales para que sean independientes del tipo de desarrollo, de las fases de este e incluso del producto.

Los tipos de defectos se asocian a una o más fases del proceso de desarrollo. De esta manera, la asociación indica donde se espera un pico en defectos de tipo funcional. Por tanto, la tabla de mapeo describe el perfil de defectos en cada una de las fases.

	Funcionales	Validación	Sincronización	Algoritmo
Diseño	X			
Diseño de bajo nivel			X	
Implementación		X		X
Inspección de alto nivel	X			
Inspección de bajo nivel			X	
Inspección del código		X		X
Pruebas unitarias		X		X
Pruebas funcionales	X			X
Pruebas del sistema			X	

El objetivo de los tipos de defectos también es poder relacionar cada tipo con algunas de las fases del proceso de desarrollo. Se trata de determinar dónde fueron inyectados los defectos en el sistema.

Relación entre tipos de defectos y fase de desarrollo:

Tipo de Defecto	**Asociación al proceso**
Función	Diseño
Interface	Diseño de bajo nivel
Chequeo	Diseño de bajo nivel o código
Asignación	Código
Serialización y temporales	Diseño de bajo nivel
Construcción / Empaquetado	Librerías
Documentación	Documentación
Algoritmo	Diseño de bajo nivel

Al concluir cada etapa de desarrollo, se elaboran gráficos que ilustran la cantidad de defectos de diversos tipos detectados durante la etapa. Comparar la frecuencia de ocurrencia de cada tipo de defecto en distintas fases proporciona una retroalimentación sobre el progreso del proceso de desarrollo. Dado que las distribuciones de defectos evolucionan con el tiempo, sirven como indicador de la madurez del producto. Cuando se detecta una desviación en el proceso a través de cambios en las curvas de distribución de defectos, el tipo de defecto identificado señala la parte del proceso que requiere atención. Los análisis causales son fundamentales en este proceso, ya que cada desarrollador contribuye con datos objetivos sobre cada defecto, los cuales se registran en una base de datos. Al término de cada fase, se lleva a cabo una reunión para analizar colectivamente los datos y determinar sus causas.

Conclusión:

El Orthogonal Defect Classification es una herramienta valiosa para la organización y el desarrollo de software para mejorar la calidad y eficiencia del proceso de desarrollo. Al proporcionar una estructura clara para la clasificación y análisis de defectos, el ODC nos ayuda a identificar áreas problemáticas y tomar las medidas correctas y efectivas contra estas.

Bibliografía:

- W&B Asset Studio. (2023, 16 noviembre). Aseguramiento de la calidad en el software: ¿qué es y qué hace? | W&B Asset Studio. Exquisite Goods. https://www.wbassetstudio.com/blog/aseguramiento-de-la-calidad-en-el-software-que-es-y-que-hace/#:~:text=clave%20de%20SQA.-,Prevenci%C3%B3n%20de%20defectos%3A%20el%20objetivo%20principal%20del%20SQA%20es%20prevenir,minimizar%20la%20probabilidad%20de%20errores.

- X. Hu and J. Liu, "Orthogonal Defect Classification-based Ontology Construction and Application of Software-hardware Integrated Error Pattern of Software-intensive Systems," 2021 23rd International Conference on Advanced Communication Technology (ICACT), PyeongChang, Korea (South), 2021, pp. 1286-1298, doi: 10.23919/ICACT51234.2021.9370747.

- Software Quality Exp. (2018, 21 marzo). What is Orthogonal Defect Classification (ODC)? By Vivek Vasudeva. Medium. https://medium.com/@SWQuality3/what-is-orthogonal-defect-classification-odc-by-vivek-vasudeva-f2e49917f478