Felipe Alejandro Jiménez Castillo
215671386

Universidad de Guadalajara - CUCEI
Computación Tolerante a Fallas
Orthogonal Defect Classification (ODC)

Contenido

1 Introducción

Uno de los retos para la medición, en el proceso de desarrollo de software, es proveer de retroalimentación rápida y efectiva hacia los desarrolladores, está es una técnica tradicional y funcional, mientras ellos entreguen reportes, pero realmente rara vez hay una visión tan clara y definida de los procesos o productos, como para tener una buena guía en la toma de decisiones. Por lo tanto esto se convierte en la toma de decisiones guiada por la intuición, más que por verdaderas medidas, análisis y procesos de ingeniería.

ODC (Orthogonal Defect Classification) hace una mejora fundamental en el nivel de tecnología disponible para asistir a las decisiones de ingeniería de software, mediante métricas y análisis. Esto es logrado mediante la explotación de defectos de software que ocurren durante todo el desarrollo y uso en el campo, capturando una cantidad significante de información. *ODC* extrae inteligentemente la información de dichos defectos y convierte lo que era semánticamente abundante en algunas métricas vitales sobre el proceso o producto.

2 Desarrollo

2.1 Métricas y software

El desarrollo de software es un proceso amorfo por naturaleza, ya sea en cascada, desarrollo iterativo o bien un proceso caótico. La definición de actividades dentro de un proceso y las transiciones entre ellas son altamente dependiente del comportamiento humano y no por restricciones físicas u obstáculos. Las pocas separaciones nítidas entre actividades se deben principalmente a la rigidez de las herramientas y sus limitaciones. Por lo tanto, las métricas en el ambiente de desarrollo de software es un gran desafió, hasta que las herramientas existen para, automáticamente, extraerlas. Conforme las tecnologías avanzan nuevas herramientas son creadas, estás directamente compiten con las actividades definidas en los procesos. Las métricas relacionadas con los defectos de software han sobrevivido esa evolución del proceso de desarrollo de software y todavía se utilizan ampliamente a pesar de la gran variabilidad en los procesos.

2.2 Defectos de Software

La palabra defecto tiende a incluirse en su propio significado, desde la perspectiva de la estricta definición, está captura las fallas, algunos errores y, a menudo, fracasos. Dado un rango especifico de significados que podrían atribuirse a un defecto, es necesario aclarar como los datos de un defecto se usaron, que aspectos son usados y con que propósito. Los defectos de software ocurren a través de todo el ciclo de vida del desarrollo de software, desde la acepción del producto, hasta el final de vida del mismo. La lengua vernácula de las organizaciones de desarrollo tiende a nombrar y tratar los defectos como objetos diferentes dependiendo del momento o el lugar en que se encuentren. Algunos de los nombres más comunes son: *bug, error, comentario, memorandum de problema del programa, problema y reporte de análisis de programa autorizado (APAR).*

2.3 Orthogonal Defect Classification

Orthogonal Defect Classification (ODC) es una técnica de software que permite a los desarrolladores comprender y mejorar la calidad del software. *ODC* funciona identificando y clasificando los defectos en el software en función de su naturaleza, ubicación y gravedad. *ODC* se basa en la idea de que los defectos no son aleatorios, sino que ocurren con mayor frecuencia en ciertos lugares del software y en ciertas etapas del proceso de desarrollo. Al identificar y clasificar los defectos, los desarrolladores pueden identificar las áreas del software que necesitan más atención.

ODC utiliza un conjunto de atributos para clasificar los defectos. Los atributos comunes incluyen:

- *Naturaleza*: El tipo de defecto, como un error de lógica, un error de interfaz o un error de rendimiento.
- *Ubicación*: La parte del software donde ocurrió el defecto.
- *Gravedad*: La severidad del defecto, como crítico, mayor o menor.

Los desarrolladores pueden utilizar ODC para identificar los siguientes tipos de patrones:

- *Patrón de ubicación*: Los defectos ocurren con mayor frecuencia en ciertas partes del software.
- *Patrón de gravedad*: Los defectos de mayor gravedad ocurren con mayor frecuencia en ciertas partes del software o en ciertas etapas del proceso de desarrollo.
- *Patrón de tiempo*: Los defectos ocurren con mayor frecuencia en ciertas etapas del proceso de desarrollo.

Al identificar estos patrones, los desarrolladores pueden tomar medidas para mejorar la calidad del software. Por ejemplo, si los defectos de mayor gravedad ocurren con mayor frecuencia en la etapa de diseño, los desarrolladores pueden dedicar más tiempo a la revisión de diseño. *ODC* es una herramienta poderosa que puede ayudar a los desarrolladores a mejorar la calidad del software. Al identificar y clasificar los defectos, los desarrolladores pueden identificar las áreas del software que necesitan más atención. Esto puede ayudar a los desarrolladores a prevenir defectos en el futuro y a crear software de alta calidad.

El proceso de ODC generalmente se lleva a cabo en las siguientes etapas:

- *Identificación de defectos*: En esta etapa, los equipos de desarrollo y prueba identifican y registran los defectos que se encuentran durante el proceso de desarrollo de software. Los defectos pueden ser errores de programación, problemas de diseño, problemas de documentación, etc.
- *Clasificación ortogonal*: En esta etapa, los defectos se categorizan en clases de acuerdo con su naturaleza. Las clases suelen ser predefinidas y representan diferentes aspectos del software. Algunas categorías comunes incluyen funcionalidad, usabilidad, rendimiento, seguridad, etc.
- *Priorización*: Una vez que los defectos están clasificados en categorías, se pueden priorizar según su gravedad y su impacto en el software. Esto ayuda a los equipos de desarrollo a centrarse en resolver los defectos más críticos primero.
- *Análisis de tendencias*: A lo largo del tiempo, el análisis de tendencias puede revelar patrones y áreas problemáticas recurrentes en el desarrollo de software. Esto permite a las organizaciones tomar medidas proactivas para abordar las causas subyacentes de los defectos.
- *Mejora continua*: ODC se utiliza como una herramienta de mejora continua en el proceso de desarrollo de software. A medida que se identifican y resuelven defectos, se espera que la calidad del software mejore con el tiempo.

Algunos beneficios de utilizar *ODC*:

- *Mejora la calidad del software*: *ODC* ayuda a los desarrolladores a identificar y corregir los defectos en el software, lo que mejora la calidad general del software.
- *Reduce los costos de desarrollo*: Al identificar y corregir los defectos temprano en el proceso de desarrollo, *ODC* puede ayudar a reducir los costos de desarrollo.
- *Mejora la satisfacción del cliente*: Los clientes están más satisfechos con el software que es de alta calidad y libre de defectos.

3 Conclusión

Orthogonal Defect Classification (ODC) se posiciona como una herramienta infalible en la gestión de la calidad del software, proporcionándonos una estructura sólida y meticulosa para la categorización de defectos. Esta estructura no solo simplifica la identificación de áreas problemáticas, sino que también brinda un marco de referencia coherente para evaluar la gravedad y la frecuencia de los defectos. La información recopilada a través del *ODC* se convierte en un recurso fundamental para la toma de decisiones informadas en el proceso de desarrollo de software. Esta información no solo ayuda a priorizar las áreas que requieren atención inmediata, sino que también respalda la planificación estratégica a largo plazo. Cuando se implementa con precisión y se integra de manera efectiva en el ciclo de desarrollo, el ODC puede catalizar una mejora continua y sostenible en la calidad del software. Esto, a su vez, puede tener un impacto positivo tanto en la satisfacción del cliente como en el éxito general de los proyectos de desarrollo, al minimizar los riesgos y optimizar los recursos disponibles.

4 Bilbiografia

1. Chillarege, R. (1996). Orthogonal defect classification. Handbook of software reliability engineering, 359-399.

2. Myers, G. J., Sandler, C., & Badgett, T. (2012). The art of software testing: Myers/art (G. J. Myers, T. Badgett, & C. Sandler, Eds.). John Wiley & Sons.

3. Black, R. (2013). Managing the testing process: Practical tools and techniques for managing hardware and software testing. Wiley.