BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

***

BÁO CÁO

Môn: Thiết kế phần mềm

The Broken Window Theory & The Boy Scout Rule

GVHD: Thầy Trần Duy Thảo
SVTH: Nguyễn Nhật Tân -22120325
Trần Nhật Tân -22120328
Lương Thị Diệu Thảo - 22120337

Thủ Đức, ngày 28 tháng 03 năm 2025

MỤC LỤC

A. THE BROKEN WINDOW THEORY

a) Định nghĩa

The Broken Window Theory (Lý thuyết cửa sổ vỡ) được James Q. Wilson và George L. Kelling đưa ra vào năm 1982. Lý thuyết này lập luận rằng những dấu hiệu nhỏ của sự hỗn loạn và suy thoái (như cửa sổ bị vỡ, rác thải bừa bãi, vẽ bậy lên tường) có thể dẫn đến tình trạng phạm tội nghiêm trọng hơn. . Nó chỉ ra rằng nếu một cửa sổ bị vỡ không được sửa chữa, nó sẽ tạo cảm giác rằng không ai quan tâm đến khu vực đó, dẫn đến sự gia tăng của hành vi tiêu cực.

b) Nguyên lý cốt lõi

Lý thuyết này dựa trên một giả định tâm lý và xã hội rằng:

- Nếu một cửa sổ bị vỡ mà không được sửa chữa, người ta sẽ có xu hướng nghĩ rằng không ai quan tâm đến tài sản này.
- Sự thờ ơ này sẽ khuyến khích những hành vi phá hoại khác, thậm chí là các tội ác nghiêm trọng hơn.
- Ngược lại, nếu môi trường được duy trì trật tự, sạch sẽ, thì hành vi phạm tội sẽ ít có cơ hội phát triển.

c) Ứng dụng trong phát triển phần mềm

Trong lĩnh vực phát triển phần mềm, lý thuyết này có thể được áp dụng theo cách sau:

1. Mã nguồn xấu sẽ dẫn đến mã nguồn tệ hơn: Nếu mã nguồn không được duy trì tốt, nó sẽ dẫn đến sự lộn xộn và khó bảo trì.
2. Khi gặp lỗi nhỏ, hãy sửa ngay: Nếu một lỗi nhỏ không được khắc phục sớm, nó có thể dẫn đến những lỗi lớn hơn.
3. Chất lượng mã nguồn ảnh hưởng đến đội ngũ phát triển: Một dự án có nhiều code xấu sẽ làm giảm động lực của lập trình viên.

d) Ứng dụng cụ thể

1. Chất lượng mã nguồn:
   - Nếu một đoạn code khó hiểu, không tuân theo coding style nhưng không được chỉnh sửa, các lập trình viên khác có thể tiếp tục viết code theo cách tương tự, làm cho toàn bộ dự án trở nên khó bảo trì.
   - Thực hiện cải tiến mã nguồn liên tục giúp giữ cho codebase sạch sẽ, dễ đọc và dễ mở rộng.
2. Quản lý lỗi:
   - Nếu các lỗi nhỏ không được sửa chữa kịp thời, chúng có thể tích lũy và gây ra những lỗi nghiêm trọng hơn trong tương lai.
   - Viết test case trước khi viết code giúp phát hiện và khắc phục lỗi từ sớm.
3. Văn hóa làm việc nhóm:

- Nếu nhóm phát triển không tuân theo tiêu chuẩn code review, không thực hiện documentation đầy đủ, thì dần dần môi trường làm việc sẽ trở nên kém chất lượng.
- Duy trì kỷ luật trong nhóm, áp dụng best practices giúp đảm bảo tính chuyên nghiệp và hiệu suất cao.

e) Kết luận:

"Thuyết cửa sổ vỡ" nhấn mạnh tầm quan trọng của việc duy trì chất lượng phần mềm ngay từ đầu. Bằng cách sửa lỗi sớm, giữ code sạch, tuân theo best practices, và duy trì văn hóa làm việc chuyên nghiệp, các nhóm phát triển có thể giảm thiểu rủi ro và đảm bảo phần mềm phát triển bền vững.

B. THE BOY SCOUT RULE

a) Định nghĩa

The Boy Scout Rule (Quy tắc hướng đạo sinh) xuất phát từ nguyên tắc của hướng đạo sinh: "Leave the campground cleaner than you found it" (Hãy để khu cắm trại sạch hơn so với khi bạn đến). Trong phát triển phần mềm, nguyên tắc này khuyến khích lập trình viên cải thiện mã nguồn mỗi khi họ làm việc với nó.

b) Tầm quan trọng

Khi lập trình viên phải làm việc với thời hạn gấp rút, họ thường ưu tiên tốc độ hoàn thành hơn giữ mã nguồn gọn gàng, dễ đọc.

Nếu không dọn dẹp thường xuyên, mã nguồn sẽ trở nên rối rắm và khó làm việc, tạo nên các nợ kỹ thuật, gây khó khăn trong việc bảo trì mã nguồn:

- Mã nguồn khó hiểu, khó cải tiến hơn.
- Tốn thời gian hiểu mã nguồn và sửa lỗi khi làm việc trong dự án, nơi mọi người làm việc trên cùng mã nguồn.
- Tốn thời gian để refactor thay vì phát triển thêm tính năng vì mã nguồn trở nên quá phức tạp.

c) Lợi ích

- Giảm nợ kỹ thuật: Giải quyết vấn đề lập tức giúp giảm những vấn đề lớn hơn trong tương lai và không phải tốn thời gian hiểu lại mã nguồn.
- Đảm bảo liên tục cải thiện mã nguồn: Thay vì đợi đến lúc phải refactor lại lượng lớn mã nguồn, lập trình viên có thể cải thiện mã nguồn từng chút nhưng thường xuyên. Điều này giúp mã nguồn luôn có thể kiểm soát được và thích ứng được khi dự án phát triển.
- Hợp tác hiệu quả hơn: Nếu mọi người đều theo quy tắc này, mã nguồn sẽ luôn sạch và dễ hiểu, giúp team dễ làm việc với nhau hơn.
- Tiết kiệm thời gian: Mã nguồn sạch giúp dễ debug và test hơn.
- Liên tục cung cấp tính năng cho khách hàng trong khi vẫn giữ mã nguồn sạch.
- Xây dựng văn hóa quan tâm và trách nhiệm với công việc.

d) Ứng dụng trong phát triển phần mềm

1. Luôn cải thiện mã nguồn: Khi sửa lỗi hoặc thêm tính năng mới, hãy làm cho mã nguồn sạch hơn, dễ hiểu hơn.
2. Tái cấu trúc mã nguồn (Refactoring): Không cần thay đổi toàn bộ hệ thống một lúc, nhưng mỗi lần làm việc, hãy cố gắng làm cho mã tốt hơn.
3. Chú ý đến quy ước lập trình: Cải thiện cách đặt tên biến, tách nhỏ hàm, tối ưu hóa hiệu suất.
4. Sử dụng tools: Một số tools có thể tự động phát hiện và sửa lỗi trong mã nguồn.