ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-------o0o-------

NGUYỄN DUY NAM

NGHIÊN CỨU JUNIT VÀ ỨNG DỤNG KIỂM THỬ TỰ ĐỘNG

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

NGÀNH KỸ THUẬT PHẦN MỀM

Thái Nguyên, tháng 05 năm 2024

ĐẠI HỌC THÁI NGUYÊN

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-------o0o-------

ĐỒ ÁN

TỐT NGHIỆP ĐẠI HỌC

NGÀNH KỸ THUẬT PHẦN MỀM

Đề tài:

NGHIÊN CỨU JUNIT VÀ ỨNG DỤNG KIỂM THỬ TỰ ĐỘNG

Sinh viên thực hiện: **NGUYỄN DUY NAM**

Lớp: **KTPM-K16B**

Giáo viên hướng dẫn: **TS. TÔ HỮU NGUYÊN**

Thái Nguyên, tháng 05 năm 2024

MỤC LỤC

DANH MỤC HÌNH ẢNH

CHƯƠNG 1: LÝ THUYẾT KIỂM THỬ PHẦN MỀM

Chương đầu tiên của báo cáo tập trung vào việc nghiên cứu và phân tích các khái niệm cơ bản của kiểm thử phần mềm. Nó cung cấp một cái nhìn tổng quan về cách phân loại kiểm thử phần mềm, trình bày các quy trình kiểm thử, mức độ khác nhau và các kỹ thuật áp dụng trong lĩnh vực kiểm thử phần mềm.

1.1. Khái niệm

Một số nhà nghiên cứu đã đưa ra các định nghĩa khác nhau về kiểm thử phần mềm như sau:

Theo Dijkstra: Kiểm thử có thể chỉ ra những lỗi đang tồn tại nhưng không thể chứng minh sự vắng mặt của những lỗi chưa được phát hiện.

Theo Beizer:

Định luật 1: Mọi phương pháp kiểm thử chỉ có thể loại bỏ một phần các lỗi.

Định luật 2: Phần mềm càng phức tạp thì càng vượt quá khả năng kiểm soát của con người.

Theo IEEE: Kiểm thử là quá trình chạy hệ thống hoặc một thành phần dưới các điều kiện cụ thể, theo dõi và ghi nhận kết quả, và đánh giá những kết quả đó để xác định chất lượng của hệ thống hay thành phần.

Theo Myer: Kiểm thử là quá trình chạy chương trình với mục tiêu phát hiện lỗi.

Nói một cách ngắn gọn, kiểm thử phần mềm là quá trình được thực hiện để cung cấp thông tin về chất lượng sản phẩm hoặc dịch vụ cho các bên liên quan. Đơn giản hơn, nó là quá trình tìm kiếm sự cố hoặc xác nhận sự chính xác của hoạt động phần mềm.

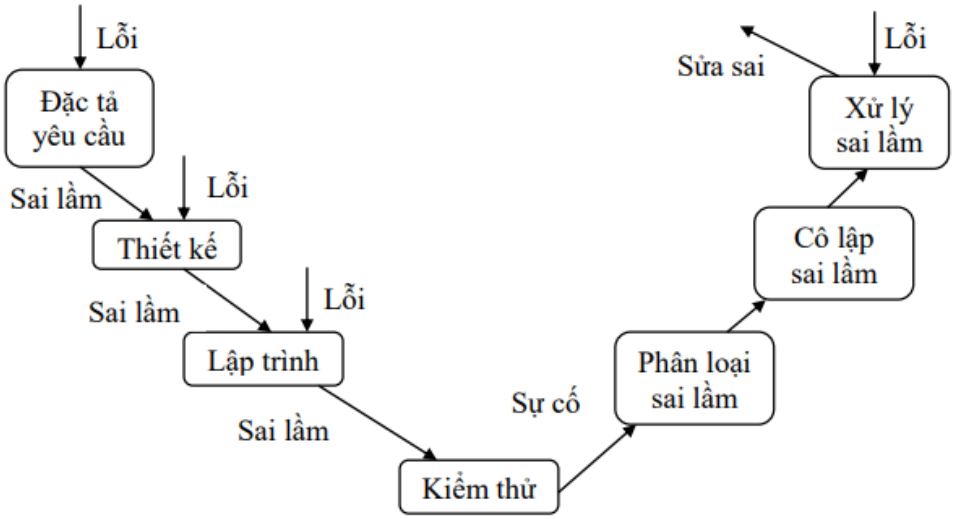
1.2. Mục tiêu của kiểm thử phần mềm

- Phát hiện các lỗi do lập trình viên gây ra trong quá trình viết code.
- Tăng cường sự tự tin và cung cấp bằng chứng về chất lượng sản phẩm
- Ngăn chặn sự xuất hiện của các lỗi.
- Đảm bảo sản phẩm phần mềm phù hợp với các yêu cầu kinh doanh và đáp ứng nhu cầu của người dùng.

1.3. Vai trò

Vai trò của kiểm thử phần mềm đóng góp trọng yếu trong việc cải thiện chất lượng và độ tin cậy của phần mềm trong suốt quá trình phát triển. Quá trình kiểm thử

bao gồm các bước "phát hiện lỗi, xác định lỗi, loại bỏ lỗi" giúp cải tiến đáng kể chất lượng của sản phẩm phần mềm. Việc xác định được mức độ hoàn thiện của sản phẩm trước khi nó được triển khai sử dụng giúp giảm thiểu tối đa các rủi ro có thể xảy ra trong giai đoạn phát triển phần mềm.



Hình 1-1: Vòng đời của quá trình kiểm thử

1.4. Nguyên tắc kiểm thử phần mềm

- Nguyên tắc 1: Kiểm thử tiết lộ sự tồn tại của lỗi.
  - Kiểm thử có khả năng chỉ ra các lỗi trong sản phẩm, nhưng không thể xác nhận rằng sản phẩm không còn chứa lỗi nào khác.
  - Dù sản phẩm đã được kiểm thử kỹ lưỡng đến mức nào đi nữa, vẫn có khả năng tồn tại lỗi.
  - Vì vậy, việc xây dựng các trường hợp kiểm thử sao cho khám phá được nhiều lỗi nhất có thể là rất quan trọng.
- Nguyên tắc 2: Kiểm thử toàn bộ là không thể.
  - Nếu sản phẩm quá đơn giản hoặc không có nhiều giá trị đầu vào, việc kiểm thử đủ để chứng minh tính ổn định của sản phẩm có thể thực hiện được. Tuy nhiên, với hầu hết các sản phẩm phần mềm hiện nay, đặc biệt là những sản phẩm phức tạp, việc kiểm thử trở nên khó khăn và thường không thể thực hiện toàn bộ. Sản phẩm phần mềm thường phát triển trên nhiều nền tảng và công nghệ, với khả năng lưu trữ và kết nối dữ liệu lớn, làm cho việc kiểm thử toàn bộ gần như là không khả thi. Mặc dù việc kiểm thử với tất cả các kết

hợp đầu vào và đầu ra, cùng với tất cả các kịch bản là không khả thi, nhưng nếu tập trung vào một số trường hợp quan trọng và ưu tiên có nguy cơ lỗi cao hơn, ta có thể đạt được kết quả tốt hơn.

- Nguyên tắc 3: Kiểm thử càng sớm càng tốt.
  - Nguyên tắc này khuyến khích việc bắt đầu thử nghiệm phần mềm từ giai đoạn đầu của quá trình phát triển.
  - Thực hiện các hoạt động kiểm thử phần mềm từ giai đoạn sớm sẽ phát hiện lỗi một cách nhanh chóng.
  - Điều này cho phép chuyển giao phần mềm đúng thời gian và chất lượng theo yêu cầu.
  - Hơn nữa, nó còn giúp giảm chi phí sửa lỗi.
- Nguyên tắc 4: Lỗi thường được phân bố tập trung.
  - Thông thường, phần lớn các lỗi tập trung vào các module hoặc thành phần chức năng chính của hệ thống. Điều này tương ứng với nguyên lý Pareto, trong đó 80% số lượng lỗi được tìm thấy trong 20% tính năng của hệ thống. Bằng cách nhận biết và xác định được điểm này, bạn có thể tập trung nỗ lực vào việc tìm kiếm lỗi trong khu vực đã xác định. Phương pháp này được coi là một trong những cách hiệu quả nhất để thực hiện kiểm thử hiệu quả.
- Nguyên tắc 5: Nghịch lý thuốc trừ sâu.
  - Nghịch lý thuốc trừ sâu trong kiểm thử phần mềm chỉ ra rằng, việc áp dụng lặp đi lặp lại cùng một bộ test cases sẽ dần giảm khả năng phát hiện lỗi mới trong hệ thống. Điều này xảy ra bởi vì khi phần mềm được cập nhật và cải thiện, các lỗi ban đầu có thể đã được sửa, trong khi đó các trường hợp kiểm thử cũ có thể không còn hiệu quả. Do đó, khi có sự thay đổi hoặc cập nhật lỗi, hoặc khi tính năng mới được thêm vào, việc thực hiện kiểm thử hồi qui trở nên thiết yếu để đảm bảo rằng những thay đổi đó không tạo ra lỗi mới hoặc ảnh hưởng tiêu cực đến các phần khác của phần mềm.
- Nguyên tắc 6: Kiểm thử phụ thuộc vào ngữ cảnh.
  - Nguyên tắc "Kiểm thử phụ thuộc vào ngữ cảnh" nhấn mạnh rằng việc áp dụng một phương pháp kiểm thử giống nhau cho các loại ứng dụng khác nhau là không phù hợp. Ví dụ, sử dụng cùng một chiến lược kiểm thử cho cả

ứng dụng web và ứng dụng di động là không thích hợp do bản chất và yêu cầu của mỗi nền tảng là khác nhau. Do đó, cần phải phát triển các chiến lược kiểm thử đặc thù phù hợp với từng loại ứng dụng cụ thể.

- Nguyên tắc 7: Quan niệm nhầm lẫn về "hết lỗi".

- Không phát hiện lỗi trên sản phẩm không có nghĩa là sản phẩm đã hoàn toàn chuẩn bị để ra thị trường.

Không phát hiện lỗi cũng có thể là kết quả của việc bộ test case chỉ được thiết kế để kiểm tra các tính năng hoạt động đúng theo yêu cầu, chứ không phải để phát hiện lỗi mới.

1.5. Phân loại kiểm thử phần mềm

Phân loại kiểm thử phần mềm bao gồm hai phương pháp chính: kiểm thử tĩnh và kiểm thử động.

1.5.1. Kiểm thử tĩnh

Kiểm thử tĩnh là một phương pháp kiểm thử phần mềm mà không đòi hỏi thực thi chương trình. Điều này đối lập hoàn toàn với kiểm thử động. Trong kiểm thử tĩnh, hoạt động chính là kiểm tra tính đúng đắn của mã nguồn, thuật toán hoặc tài liệu mà không cần chạy chương trình. Thường thì kiểm thử tĩnh được thực hiện bởi lập trình viên trong quá trình phát triển phần mềm. Một ưu điểm của kiểm thử tĩnh là việc phát hiện lỗi sớm giúp giảm thiểu chi phí sửa chữa so với việc lỗi được phát hiện ở giai đoạn sau.

1.5.2 Kiểm thử động

Kiểm thử động liên quan đến việc thực thi chương trình để phát hiện lỗi và các vấn đề liên quan đến hiệu suất của hệ thống. Trong quá trình kiểm thử động, chương trình được chạy với các dữ liệu đầu vào để kiểm tra xem nó hoạt động đúng đắn hay không. Tuy nhiên, do không thể thử nghiệm tất cả các trường hợp có thể xảy ra, nên ta thường chọn một tập hợp con của dữ liệu đầu vào để thực thi. Quá trình này tạo ra các ca kiểm thử để đảm bảo rằng chương trình hoạt động như mong đợi trong các điều kiện khác nhau.

Trong kiểm thử động, có hai kỹ thuật chính là kiểm thử hộp trắng (hay kiểm thử cấu trúc) và kiểm thử hộp đen (hay kiểm thử chức năng).

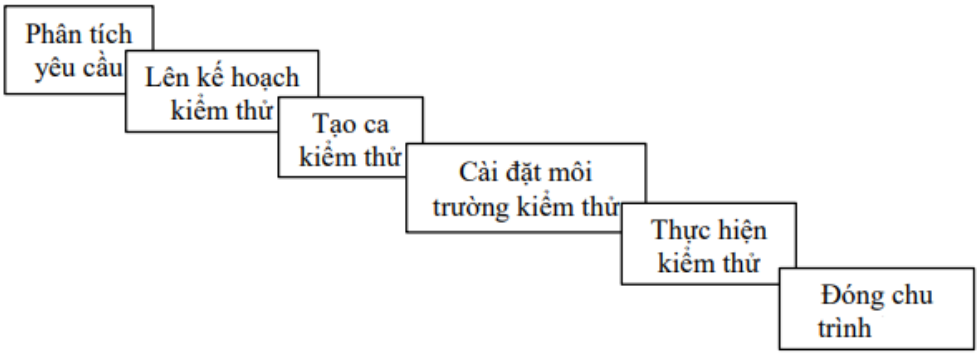
Kiểm thử hộp trắng: Đây là kỹ thuật kiểm thử dựa vào cấu trúc nội bộ của chương trình, như thuật toán và cấu trúc mã nguồn, nhằm đảm bảo rằng tất cả các dòng lệnh và điều kiện trong mã nguồn được thực thi ít nhất một lần. Người kiểm thử sẽ trực tiếp tiếp cận và phân tích mã nguồn để tạo ra các ca kiểm thử. Các kỹ thuật phổ biến của kiểm thử hộp trắng bao gồm bao phủ dòng lệnh, bao phủ nhánh và bao phủ đường dẫn. Tuy nhiên, kỹ thuật này có nhược điểm là không thể đảm bảo rằng chương trình tuân thủ đúng theo các đặc tả, và khó phát hiện các lỗi do dữ liệu hoặc các thiếu sót về đường dẫn. Một số công cụ phổ biến được sử dụng cho kiểm thử hộp trắng bao gồm Veracode, RCUNIT và GoogleTest.

Kiểm thử hộp đen: Đây là kỹ thuật kiểm thử dựa trên các đầu vào và đầu ra của chương trình mà không quan tâm đến cấu trúc nội bộ của mã nguồn. Trong kỹ thuật này, người kiểm thử xác định và tạo ra các nhóm giá trị đầu vào để thực thi các chức năng của chương trình. Các kỹ thuật chính của kiểm thử hộp đen bao gồm phân tích giá trị biên, phân vùng tương đương và lập bảng quyết định.

1.6. Quy trình kiểm thử phần mềm

Quy trình kiểm thử phần mềm không chỉ là một hoạt động đơn lẻ mà là một loạt các giai đoạn được thực hiện với sự phối hợp của nhiều bên liên quan. Mục tiêu của quy trình này là làm rõ các công đoạn, bước kiểm thử, người chịu trách nhiệm và thời điểm nào kiểm thử sẽ được thực hiện trong toàn bộ quá trình phát triển phần mềm

Các giai đoạn trong quy trình kiểm thử phần mềm có thể khái quát như sau:



Hình 1-2: Quy trình kiểm thử phần mềm

1.6.1. Phân tích yêu cầu

Trong quá trình phân tích yêu cầu, nhóm kiểm thử thường tiến hành nghiên cứu tài liệu yêu cầu để hiểu rõ về phần mềm và đưa ra cái nhìn tổng quan về dự án. Mục