BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Nguyễn Mạnh Tiền

Giảng viên hướng dẫn: ThS. Nguyễn Trịnh Đông

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

XÂY DỰNG CA KIỂM THỬ TỪ BIỂU ĐỒ LUỒNG DỮ LIỆU

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

NGÀNH: CÔNG NGHỆ THÔNG TIN

Sinh viên : Nguyễn Mạnh Tiền

Giảng viên hướng dẫn : ThS. Nguyễn Trịnh Đông

HẢI PHÒNG - 2018

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: Nguyễn Mạnh Tiền **Mã SV: 1412101135**

Lớp: CT1801 **Ngành: Công nghệ thông tin**

Tên đề tài: Xây dựng ca kiểm thử từ biểu đồ luồng dữ liệu

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy giáo, Ths. Nguyễn Trịnh Đông – giảng viên khoa CNTT đã tận tâm và nhiệt tình hướng dẫn, dạy bảo trong suốt quá trình học tập và làm đồ án tốt nghiệp. Với sự chỉ bảo của thầy, em đã có những định hướng tốt trong việc triển khai và thực hiện các yêu cầu trong quá trình làm đồ án tốt nghiệp. Em xin chân thành cảm ơn sự dạy bảo và giúp đỡ của các thầy, cô giáo Khoa Công nghệ thông tin – Trường Đại học Dân lập Hải Phòng đã trang bị cho em những kiến thức cơ bản nhất để em có thể hoàn thành tốt bài báo cáo này. Do khả năng và thời gian còn hạn chế, kinh nghiệp làm việc thực tế chưa nhiều nên không tránh khỏi những thiếu sót. Em rất mong nhận được sự chỉ bảo của các thầy cô và các bạn. Cuối cùng em xin được gửi tới các thầy, cô và toàn thể các bạn lời chúc sức khỏe, thành thông. Chúc các thầy cô đạt được nhiều thành tựu trong sự nghiệp trồng người.

Em xin chân thành cảm ơn!

Hải Phòng, ngày 30 tháng 3 năm 2018

Sinh viên

Nguyễn Mạnh Tiền

MỤC LỤC

DANH MỤC HÌNH VẼ BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT

Stt	Tên viết tắt	Tên đầy đủ	Ý nghĩa
1	IEEE	Institute of Electrical and Electronics Engineers.	Viện kỹ nghệ điện và điện tử.
2	CEF	Common European Framework.	Là phương thức chuyển mạch do Cisco phát triển áp dụng cho các dòng Multiplayer Switch và Router của hãng.
3	WPF	Windows Presentation Foundation	Là công nghệ kế tiếp Windows Form dùng để xây dựng các ứng dụng dành cho máy trạm chạy hệ điều hành Windows.
4	SAP	System Application Programing.	Là chương trình hệ thống dành cho các doanh nghiệp do IBM phát triển.
5	.NET	.NET Framework	Là một nền tảng lập trình và cũng là một nền tảng thực thi ứng dụng chủ yếu trên hệ điều hành Microsoft Windows được phát triển bởi Microsoft.

MỞ ĐẦU

Phần mềm đóng một vai trò quan trọng trong mọi lĩnh vực của cuộc sống. Trong đó, kiểm thử phần mềm là một trong những quy trình đảm bảo phần mềm hoạt động chính xác theo yêu cầu của thiết kế. Do đó, việc nắm vững kiến thức và rèn luyện các kỹ năng về kiểm thử phần mềm là một tiêu chí quan trọng đối với sinh viên ngành Công nghệ Thông tin.

Quy trình kiểm thử phần mềm được chia thành nhiều giai đoạn và nhiều hoạt động khác nhau tùy thuộc vào phần mềm được phát triển dựa trên các quy trình khác nhau. Dù phần mềm được phát triển theo quy trình nào thì các bước kiểm thử đều có những giai đoạn giống nhau gồm kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống, v.v. Các hoạt động của kiểm thử được tiến hành từ những giai đoạn đầu của quá trình phát triển phần mềm. Căn cứ vào bản đặc tả yêu cầu phần mềm, người ta có thể xây dựng các ca kiểm thử và dựa vào đó khi triển khai phần mềm đến đâu thì hoạt động kiểm thử phần mềm được thực hiện ngay đến đó để kịp thời phát hiện lỗi trong sản phẩm phần mềm. Khóa luận này, với tên đề tài ***"Phương pháp tính toán các ca kiểm thử dựa trên biểu đồ luồng dữ liệu"***, lần lượt trình bày một số khái niệm cơ bản về phần mềm, kiểm thử phần mềm, các bước xác định ca kiểm thử từ biểu đồ luồng dữ liệu, và sử dụng công cụ Ranorex Studio trong kiểm thử phần mềm. Nội dung của khóa luận được trình bày theo cấu trúc dưới đây.

Chương 1: Các khái niệm cơ bản

Chương này cung cấp các kiến thức cơ bản trong lĩnh vực phát triển phần mềm và kiểm thử phần mềm như các khái niệm về phần mềm, lỗi phần mềm, quy trình xử lí lỗi phần mềm và khái niệm cơ bản trong kiểm thử phần mềm.

Chương 2: Xây dựng ca kiểm thử từ biểu đồ luồng dữ liệu

Chương này trình bày các bước phân tích biểu đồ luồng dữ liệu, trên cơ sở đó xây dựng ca kiểm thử và các thành phần liên quan.

Chương 3: Thực nghiệm với công cụ Ranorex Studio

Ranorex Studio là một công cụ mạnh trong kiểm thử phần mềm cho ứng dụng Window Forms. Chương này trình bày các bước thực hiện kiểm thử các ứng dụng với công cụ Ranorex.

Kết luận:

Phần này khóa luận đưa ra những kết quả, hạn chế và hướng phát triển tiếp theo của đề tài trong tương lai.

CHƯƠNG 1: KIẾN THỨC CƠ BẢN

Chương này khóa luận trình bày những thuật ngữ và khái niệm cơ bản về phần mềm, lỗi phần mềm, xử lý lỗi phần mềm và kiểm thử phần mềm.

1.1 Khái niệm cơ bản về phần mềm

Theo IEEE (1991): Phần mềm là các chương trình máy tính kết với các dữ liệu hoặc các tài liệu hướng dẫn, đặc tả, v.v. thường gồm 4 phần được mô tả dưới đây:

- *Chương trình máy tính:* Thành phần này giúp cho máy tính thực thi các ứng dụng được yêu cầu.
- *Quy trình:* Là thành phần xác định trình tự và kế hoạch trong đó các chương trình được thực hiện, phương pháp sử dụng và những người chịu trách nhiệm cho các hoạt động của kế hoạch.
- *Các tài liệu:* Có rất nhiều những tài liệu cần thiết với nhân viên phát triển, người sử dụng và nhân viên bảo trì như: tài liệu thiết kế, tài liệu hướng dẫn sử dụng, tài liệu hướng dẫn bảo trì.
- *Dữ liệu:* Dữ liệu bao gồm tham số, mã nguồn và các danh sách thích ứng của phần mềm dành riêng cho người dùng cụ thể là dành cho hoạt động phần mềm.

1.1.1 Vòng đời phần mềm

Mô hình vòng đời phát triển phần mềm là một loạt các pha (giai đoạn) mà phần mềm trải qua từ bắt đầu phát triển đến khi phần mềm bị loại bỏ hoàn toàn. Mô hình vòng đời phát triển phần mềm thường gồm những pha sau:

- *Pha yêu cầu:* là pha đầu tiên trong quá trình xây dựng phần mềm, nhằm xác định yêu các yêu cầu phải có trong phần mềm giữa nhóm phát triển và khách hàng.

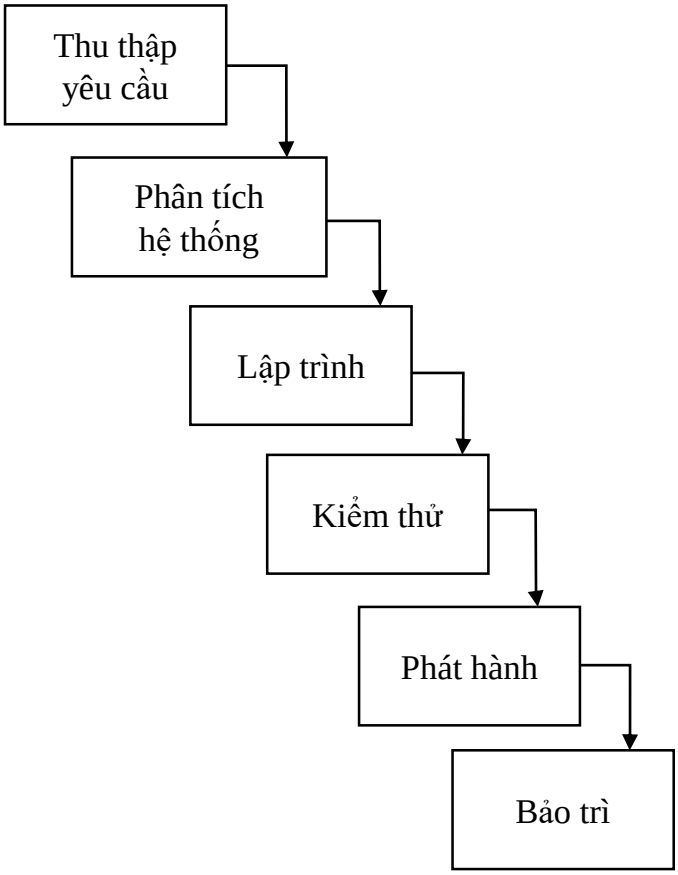
- *Pha phân tích:* Pha này phân tích các yêu cầu của khách hàng được mô tả chi tiết kết quả đầu ra, quá trình phát triển phần mềm dưới dạng "tài liệu đặc tả".
- *Pha thiết kế:* Pha này căn cứ vào tài liệu đặc tả để mô tả chi tiết cách phần mềm thực hiện các công việc cụ thể.
- *Pha lập trình:* Pha này là quá trình thực hiện viết chương trình bằng một ngôn ngữ cụ thể.
- *Pha kiểm thử hệ thống:* Pha này hoạt động sau khi giai đoạn lập trình kết thúc. Mục đích chính là phát hiện lỗi phần mềm càng nhiều càng tốt để đạt được chất lượng phần mềm ở mức chấp nhận được sau khi chỉnh sửa.
- *Pha bảo trì và loại bỏ:* Pha này bảo trì sửa lỗi có thể còn xuất hiện trong chương trình sau khi cài đặt cho khách hàng và cập nhật sửa đổi phần mềm theo ý khách hàng hoặc thích nghi với điều kiện ràng buộc để nâng cao hiệu quả làm việc. Nếu chi phí bảo trì quá lớn có thể dẫn đến loại bỏ phần mềm.

1.1.2 Quy trình phát triển phần mềm

Phần mềm được phát triển dựa trên các mô hình để xác định các hoạt động và quy trình theo trình tự nhất định. Một số mô hình phát triển phần mềm tiêu biểu sau:

a. Mô hình thác nước

Mô hình thác nước hay còn gọi là mô hình vòng đời truyền thống do tác giả Royce đề xuất năm 1970. Mô hình này yêu cầu tiếp cận một cách hệ thống, tuần tự và chặt chẽ đối với việc phát triển phần mềm, bắt đầu ở mức hệ thống và tiến dần xuống phân tích, thiết kế, mã hóa, kiểm thử, và bảo trì (Hình 1-1) [2].



Hình 1-1. Mô hình thác nước

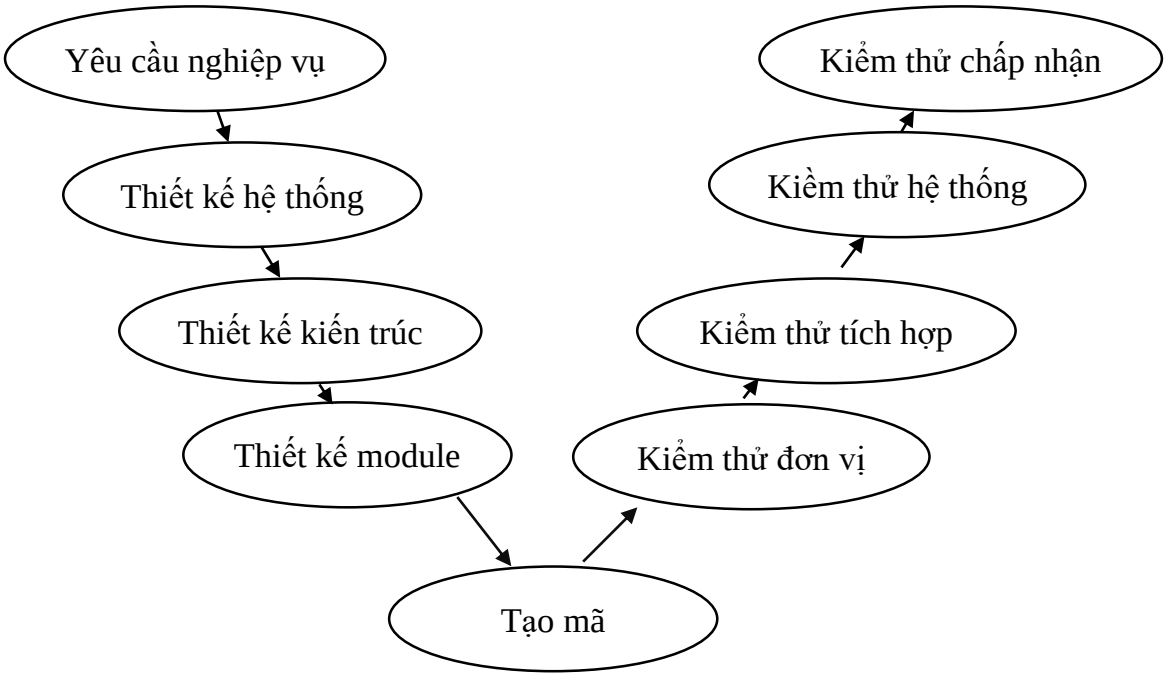
Mô hình thác nước là mô hình áp dụng theo tính tuần tự của giai đoạn phát triển phần mềm. Giai đoạn sau chỉ được thực hiện khi giai đoạn trước được hoàn thành. Đặc điểm của phát triển phần mềm mô hình thác nước được trình bày trong Hình 1-2 sau:

Ưu điểm	Nhược điểm
Dễ sử dụng, dễ tiếp cận	Khó quay lại các giai đoạn trước
Các giai đoạn và hoạt động được xác định chi tiết	Ít tính linh hoạt và phạm vi điều chỉnh khó khăn, tốn kém
Xác nhận ở từng giai đoạn	

Hình 1-2. Ưu nhược điểm phát triển mô hình thác nước

b. Mô hình chữ V

Ngoài phát triển phần mềm theo mô hình thác nước, thì mô hình chữ V cũng khá phổ biến trong các dự án phát triển phần mềm của các công ty.



Hình 1-3. Mô hình chữ V

Mô hình chữ V được thực hiện từ trái qua phải, mô tả dãy các hoạt động và kiểm thử cơ bản. Mô hình này nêu bật các mức kiểm thử liên quan đến các pha phát triển phần mềm tương ứng từng giai đoạn.

Đặc điểm phát triển phần mềm theo mô hình chữ V được trình bày trong Bảng 1-2 sau:

Ưu điểm

- Quá trình phát triển và quy trình quản lý có tính tổ chức và hệ thống
- Hoạt động tốt cho các dự án có quy mô vừa và nhỏ.
- Kiểm tra bắt đầu từ khi bắt đầu phát triển vì vậy sự mơ hồ được xác định ngay từ đầu.
- Dễ dàng quản lý vì mỗi giai đoạn có các mục tiêu và mục tiêu được xác định rõ ràng.

Nhược điểm

- Không thích hợp cho các dự án lớn và phức tạp
- Không phù hợp nếu các yêu cầu thường xuyên thay đổi.
- Không có phần mềm làm việc được sản xuất ở giai đoạn trung gian.
- Không có điều khoản cho việc phân tích rủi ro nên có sự không chắc chắn và có tính rủi ro.

1.2 Chất lượng và đảm bảo chất lượng phần mềm

1.2.1 Chất lượng phần mềm

Theo IEEE (1991):

1. Chất lượng phần mềm là mức độ mà một hệ thống, thành phần hệ thống hoặc quy trình đáp ứng được đặc tả yêu cầu.
2. Chất lượng phần mềm là mức độ mà hệ thống, thành phần hệ thống hoặc quy trình, đáp ứng được mong đợi của khách hàng hoặc người sử dụng.

Theo Pressman: Chất lượng phần mềm là phần mềm phải đáp ứng được ba yêu cầu sau:

- Các yêu cầu chức năng rõ ràng là nhân tố chính quyết định chất lượng đầu ra của sản phẩm.
- Các tiêu chuẩn chất lượng phần mềm sẽ được nói đến trong hợp đồng.
- Các đặc tính cần được đáp ứng trong quá trình phát triển cho dù không được nói đến rõ ràng trong hợp đồng.

1.2.2 Đảm bảo chất lượng phần mềm

Để đánh giá được chất lượng một phần mềm ngay cả khi nó đang hoạt động là rất khó, những người khác nhau có thể đánh giá về nó khác nhau.

Theo tác giả Daniel Galin, việc đảm bảo chất lượng phần mềm là một tập hợp các hành động cần thiết được lên kế hoạch một cách hệ thống để cung cấp đầy đủ niềm tin rằng quá trình phát triển phần mềm phù hợp để thành lập các yêu cầu chức năng kỹ thuật cũng như các yêu cầu quản lý theo lịch trình và hoạt động trong giới hạn ngân sách.

Tuy nhiên, khi đánh giá phần mềm người ta thường đưa ra một số tiêu chí để nói đến chất lượng tổng thể của phần mềm bao gồm:

- *Đạt được các mục tiêu thiết kế đề ra của tổ chức:* thực hiện được các chức năng thiết kế cho tổ chức.
- *Chi phí vận hành là chấp nhận được:* chi phí không quá cao so với lợi ích mang lại.
- *Đáp ứng được các chuẩn mực của một hệ thống tin:* đưa ra thông tin kịp thời, phù hợp với cho tính sẵn sàng hay kết quả đưa ra đúng chuẩn với mẫu bảng biểu, số chỉ tiêu, v.v.
- *Sản phẩm tạo ra có giá trị xác đáng:* thông tin đưa ra có ý nghĩa thiết thực đối với chức năng và quản lý, góp phần nâng cao chất lượng sản phẩm và dịch vụ của tổ chức.
- *Bảo trì được:* dễ bảo trì, bảo trì không quá tốn kém.
- *Khả dụng:* dễ học và dễ sử dụng.
- *Mềm dẻo – có khả năng làm thích nghi được:* có thể kiểm tra, mở rộng ứng dụng và phát triển tiếp.
- *Có tính khả chuyển:* có thể chuyển từ môi trường làm việc này sang môi trường làm việc khác.

1.3 Lỗi phần mềm

Lỗi phần mềm là sự không khớp giữa chương trình và đặc tả của nó. Lỗi phần mềm được chia thành 3 dạng:

- ***Lỗi sai:*** sản phẩm phần mềm được xây dựng khác với đặc tả.

- ***Lỗi thiếu****:* các yêu cầu cầu sản phẩm phần mềm đã có trong đặc tả nhưng lại không có trong sản phẩm thực tế.
- ***Lỗi thừa:*** phần mềm trong thực tế có những tính năng không có trong tài liệu đặc tả.

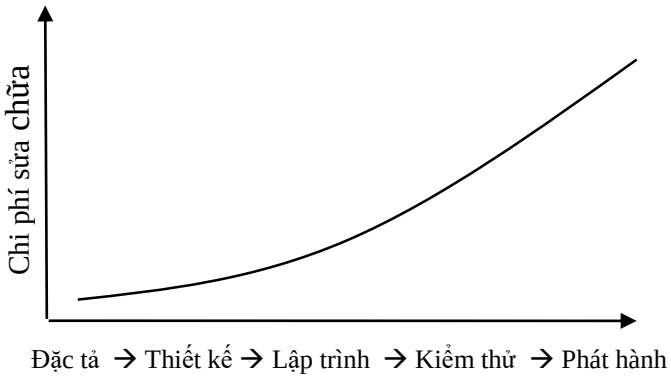
1.3.1 Nguyên nhân gây lỗi phần mềm:

Lỗi phần mềm có thể đến từ nhiều nguyên nhân khác nhau, trong đó có cả nguyên nhân chủ quan và các nguyên nhân khách quan. Sau đây là một số nguyên nhân chủ yếu gây ra lỗi phần mềm:

- *Lỗi trong giao tiếp giữa khách hàng và nhóm phát triển:* nhưng lỗi này thường xuất hiện ở đầu dự án do hiểu lầm trong giao tiếp giữa nhóm phát triển và khách hàng khi trình bày bằng lời nói và tài liệu không thống nhất.
- *Các lỗi thiết kế lôgíc:* lỗi xảy ra trong quá trình các chuyên gia thiết kế hệ thống, kỹ sư phần mềm, các nhà phân tích bỏ sót hay áp dụng thuật toán sai dẫn đến xây dựng phần mềm sai theo lôgíc.
- *Các lỗi lập trình:* có rất nhiều lý do dẫn đến việc các lập trình viên gây ra các lỗi lập trình. Ví dụ như: sai sót trong ngôn ngữ lập trình, hiểu sai các tài liệu thiết kế, v.v.
- *Các lỗi về tài liệu:* các lỗi về tài liệu là vấn đề của nhóm phát triển và bảo trì khi có những sai sót trong các tài liệu liên quan.

1.3.2 Chi phí cho việc sửa lỗi phần mềm

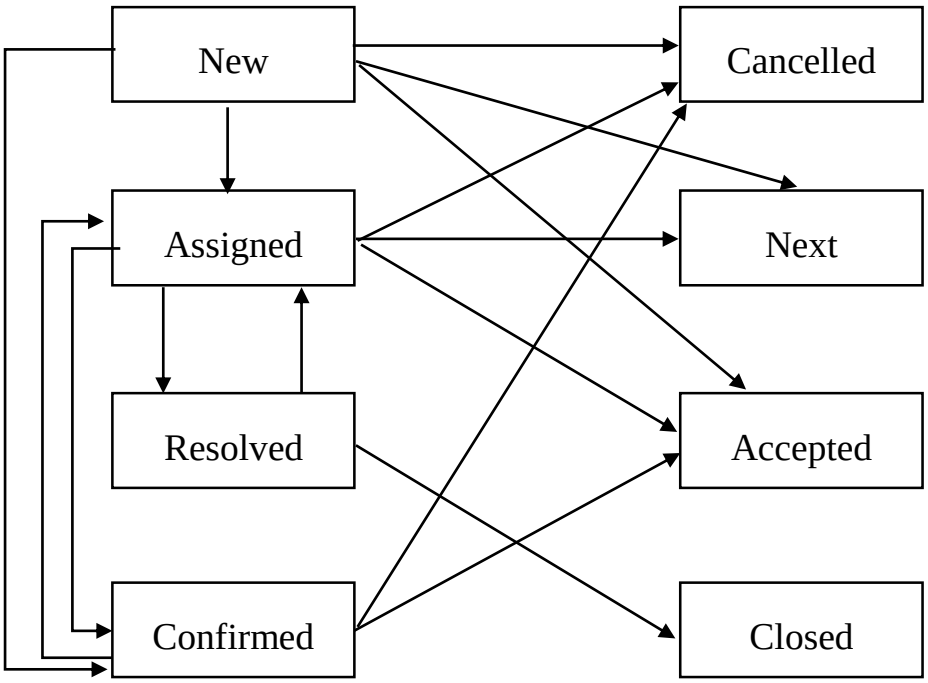
Việc sửa lỗi phần mềm có thể thực hiện trong bất cứ giai đoạn nào của vòng đời phần mềm. Tuy nhiên công việc này càng thực hiện sớm càng tốt vì càng về sau của giai đoạn sau của phần mềm thì chi phí, thời gian cho việc tìm và sửa lỗi càng tăng. Theo tài liệu của Boehm, chi phí tìm và sửa lỗi phần mềm tăng theo hàm mũ trong biểu đồ sau:



Hình 1-4. Chi phí tìm và sửa lỗi phần

1.3.3 Quy trình xử lý lỗi phần mềm

Trước khi giới thiệu về quy trình xử lý lỗi phần mềm, khóa luận trình bày về các trạng tháng có thể có của lỗi.



Hình 1-5.Trạng thái của lỗi

Trong đó, các thành phần được giải thích dưới đây.

- New: Lỗi mới.
- Assigned: Lỗi đã được gán cho nhân viên phát triển.
- Resolved: Lỗi đã được xử lý.
- Confirmed: Lỗi cần được chứng thực.
- Canceled: Lỗi được xác định không phải là lỗi, lỗi bỏ qua được

- Next: Lỗi không thuộc phạm vi của dự án hoặc sẽ được xử lý trong giai đoạn khác của dự án
- Accepted: Các lỗi có thể chấp nhận được.
- Closed: Trạng thái đóng, lỗi đã được sửa thành công.

Mỗi nhóm phát triển phần mềm sẽ sử dụng một công cụ quản lý lỗi riêng, nhưng gần như phần đa đều có một quy trình xử lý lỗi phần mềm chung. Theo đó, quy trình xử lý lỗi có thể bao gồm 6 bước chính:

Bước 1: Phát hiện phần mềm có lỗi.

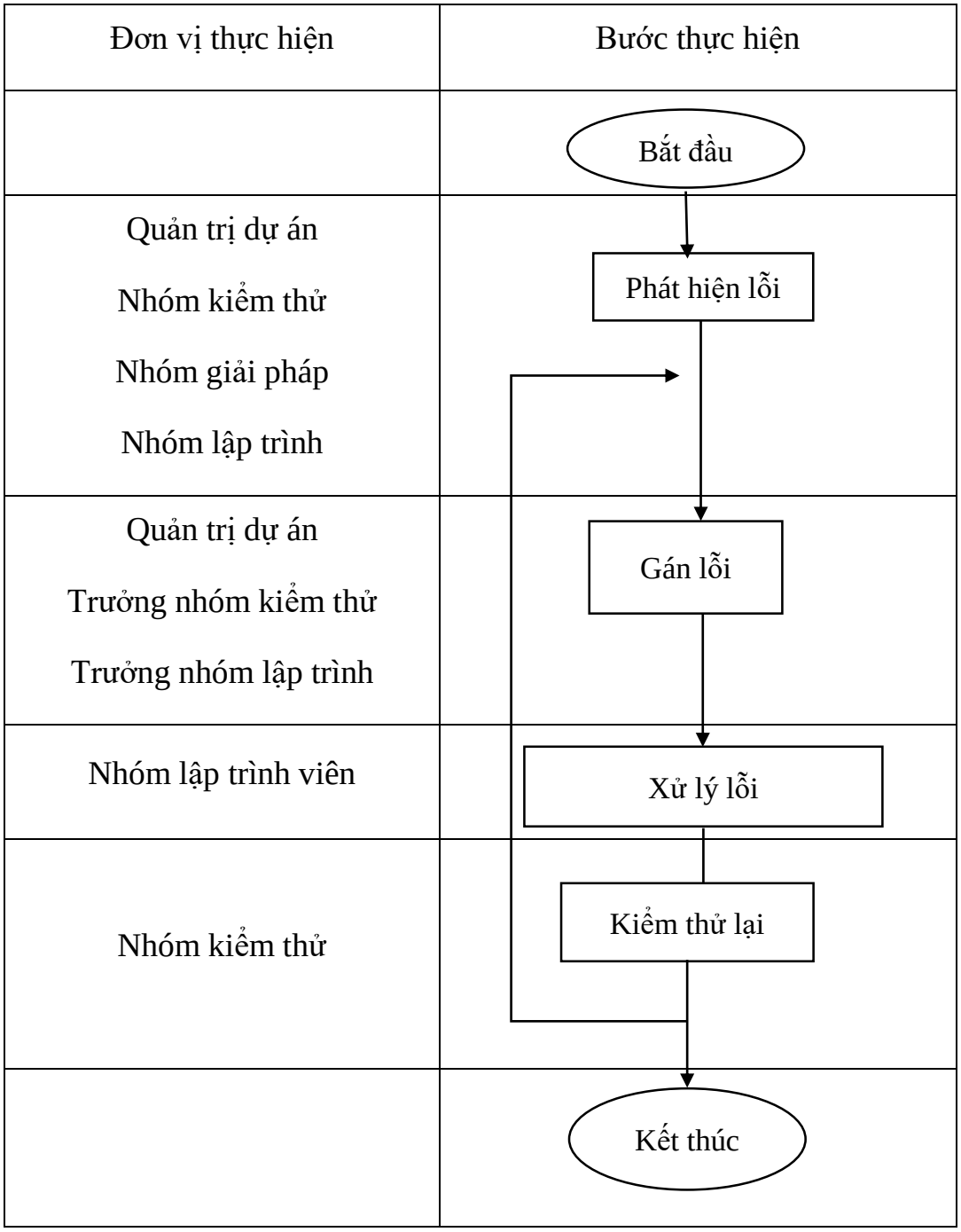
Bước 2: Đưa lỗi lên hệ thống quản lý lỗi.

Bước 3: Gán lỗi cho nhân viên phát triển.

Bước 4: Xử lý lỗi.

Bước 5: Kiểm thử lại.

Bước 6: Đóng lỗi.



1.4 Các thuật ngữ và khái niệm kiểm thử phần mềm

Tăng năng suất kiểm thử là một nhu cầu thiết yếu để tăng chất lượng phần mềm. Vì thế nghiên cứu để phát triển các kỹ thuật, công cụ kiểm thử hữu hiệu là đóng góp thiết thực nhất để tăng cường chất lượng sản phẩm phần mềm.

1.4.1 Các thuật ngữ

Kỹ thuật kiểm thử phần mềm đã phát triển và tiến hóa hàng mấy chục năm nên các thuật ngữ trong các tài liệu khác nhau thường không thống nhất và thiếu tương thích. Các thuật ngữ được trình bày trong khóa luận này dựa vào các chuẩn được viện phát triển IEEE.

Lỗi (Error): Lỗi là những vấn đề con người mắc phải trong quá trình phát triển các sản phẩm phần mềm. Khi lập trình viên phạm lỗi trong lập trình, các lỗi đó là bug(con bọ). Lỗi có thể phát tán. Lỗi là nguyên nhân dẫn đến sai.

Sai (Fault): Sai là kết quả của lỗi. Sai về nhiệm vụ khi xuất hiện khi vào sai thông tin hoặc sai về bỏ quên xuất hiện khi không vào đủ thông tin. Loại sai thứ hai khó phát hiện và sửa hơn loại sai thứ nhất.

Thất bại (Failure): Thất bại xuất hiện khi một lỗi được thực thi, nó chỉ xuất hiện dưới dạng có thể chạy được mà thông thường là mã nguồn.

Sự cố (Incident): Khi thất bại xuất hiện, nó có thể hiện thị hoặc không, tức là rõ ràng hoặc không rõ ràng đối với những người dùng hoặc kiểm thử viên. Sự cố là triệu chứng liên kết với một thất bại và thể hiện cho người dùng hoặc kiểm thử viên về sự xuất hiện của thất bại này.

Kiểm chứng và thẩm định: Kiểm chứng và thẩm định là hai khái niệm hay sử dụng nhầm lẫn, nhưng thực ra chúng có ý nghĩa khác nhau. Kiểm chứng là quá trình đảm bảo rằng một số sản phẩm phần mềm thỏa mãn các đặc tả. Còn thẩm định là quá trình để đảm bảo rằng sản phẩm đáp ứng được yêu cầu của người dùng.

1.4.2 Khái niệm kiểm thử phần mềm

Cùng với phát triển phần mềm, kiểm thử phần mềm cũng có nhiều định nghĩa khác nhau được phát biểu bởi nhiều tổ chức hay cá nhân khác nhau. Khóa luận sẽ trình bày một số định nghĩa nổi bật:

Theo Myer (1979, Chương 10)**: "**Kiểm thử là quá trình thực thi một chương trình với mục đích tìm ra lỗi." Theo như định nghĩa này, quá trình kiểm thử bao gồm tất cả các hoạt động từ kiểm tra mã nguồn đến chạy thử chương trình.

Theo IEEE Std 610.12 (IEEE, 1990)**:** Kiểm thử phần mềm là quá trình phân tích các yếu tố phần mềm để phát hiện những khác biệt giữa chương trình với các điều kiện yêu cầu và đánh giá các đặc điểm của các yếu tố phần mềm.

Theo Daniell Galin: Kiểm thử phần mềm là một quá trình được tiến hành bởi một nhóm chuyên viên kiểm thử, trong đó một đơn vị phần mềm, một nhóm các đơn vị được tích hợp, hoặc cả gọi phần mềm được kiểm tra bằng cách chạy chương trình trên máy tính. Tất cả các bước kiểm tra liền được tiến hành theo các quy trình kiểm thử và được thông qua ca kiểm thử.

1.4.3 Mục tiêu của kiểm thử phần mềm

Cũng giống như các sản phẩm máy móc và các hệ thống vật lý, mục đích của kiểm thử phần mềm là để đảm bảo hệ thống phần mềm có thể làm việc tốt như mong muốn khi chúng được đem ra thị trường tới tay khách hàng và người sử dụng. Cách thường dùng để đưa ra những kiểm chứng về chất lượng cho sản phẩm là đưa sản phẩm vào "chạy thử" hay được kiểm tra trong phòng thí nghiệm trước khi phân phối sản phẩm ra thị trường. Trong ngành Công Nghệ Phần Mềm, các sản phẩm phần mềm được kiểm tra, chạy thử được gọi chung là kiểm thử phần mềm.

Theo Daniel Galin:

Mục tiêu trực tiếp: Kiểm thử phần mềm là phát hiện và xác định càng nhiều lỗi càng tốt ở các phần mềm được kiểm thử. Tiến hành sửa lỗi ở các phần mềm được kiểm thử và kiểm thử lại cho đến khi đảm bảo các yêu cầu chất lượng phần mềm. Kiểm thử là hoạt động cần thiết và hiệu quả trong kế hoạch và ngân quỹ giới hạn.

Mục tiêu gián tiếp: Kiểm thử phần mềm nhằm biên dịch bản ghi các lỗi để nhằm mục đích phòng ngừa và khắc phục lỗi.

1.5 Nguyên tắc kiểm thử phần mềm

Để kiểm thử đạt hiệu quả thì khi tiến hành kiểm thử phần mềm cần phải tuân thủ một số quy tắc sau:

Quy tắc 1: Kiểm thử đưa ra lỗi

Kiểm thử có thể cho thấy rằng phần mềm đang có lỗi, nhưng không thể chứng minh rằng phần mềm không có lỗi. Kiểm thử được thực hiện bằng những kĩ thuật khác nhau. Kiểm thử làm giảm xác suất lỗi chưa tìm thấy vẫn còn trong phần mềm, ngay cả khi đã kiểm thử nghiêm ngặt phần mềm vẫn có thể còn lỗi. Vì vậy chúng ta phải tìm được càng nhiều lỗi càng tốt.

Quy tắc 2: Kiểm thử cạn kiệt là không thể

Nguyên tắc này nói rằng kiểm tra mọi thứ trong phần mềm một cách trọn vẹn là không thể. Kiểm thử với tất cả các kết hợp đầu vào và đầu ra, với tất cả các kịch bản là không thể trừ phi nó chỉ bao gồm ít trường hợp thì có thể kiểm thử toàn bộ. Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên sự mức độ ưu tiên chúng ta có thể tập trung việc kiểm thử vào một số điểm cần thiết, có nguy cơ lỗi cao hơn.

Quy tắc 3: Kiểm thử càng sớm càng tốt

Nguyên tắc này yêu cầu bắt đầu thử nghiệm phần mềm trong giai đoạn đầu của vòng đời phát triển phần mềm. Các hoạt động kiểm thử phần mềm từ giai đoạn đầu sẽ giúp phát hiện bug sớm hơn. Nó cho phép chuyển giao phần mềm theo yêu cầu đúng thời gian với chất lượng dự kiến.

Quy tắc 4: Sự tập trung của lỗi

Thông thường, lỗi tập trung vào những module, thành phần chức năng chính của hệ thống. Nếu xác định được điều này bạn sẽ tập trung vào tìm kiếm

lỗi quanh khu vực được xác định. Nó được coi là một trong những cách hiệu quả nhất để thực hiện kiểm tra hiệu quả.

Quy tắc 5: Nghịch lí thuốc trừ sâu

Nếu bạn sử dụng cùng một tập hợp các trường hợp kiểm thử liên tục, sau đó một thời gian các trường hợp kiểm thử không tìm thấy lỗi nào mới. Hiệu quả của các trường hợp kiểm thử bắt đầu giảm xuống sau một số lần thực hiện, vì vậy luôn luôn chúng ta phải luôn xem xét và sửa đổi các trường hợp kiểm thử trên một khoảng thời gian thường xuyên.

Quy tắc 6: Kiểm thử phụ thuộc vào ngữ cảnh

Theo nguyên tắc này thì việc kiểm thử phụ thuộc vào ngữ cảnh và chúng ta phải tiếp cận kiểm thử theo nhiều ngữ cảnh khác nhau.

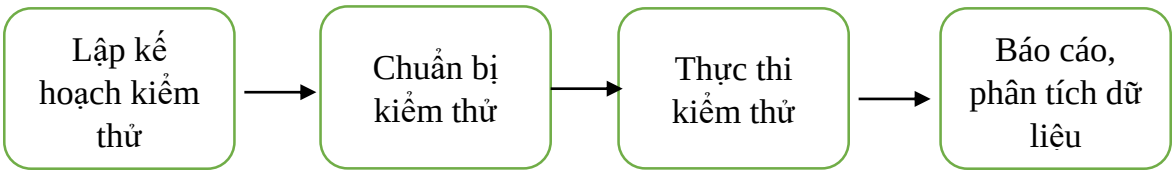
Nếu bạn đang kiểm thử ứng dụng web và ứng dụng di động bằng cách sử dụng chiến lược kiểm thử giống nhau, thì đó là sai. Chiến lược để kiểm thử ứng dụng web sẽ khác với kiểm thử ứng dụng cho thiết bị di động của Android.

Quy tắc 7: Không có lỗi - Sai lầm

Việc không tìm thấy lỗi trên sản phẩm không đồng nghĩa với việc sản phẩm đã sẵn sàng để tung ra thị trường. Việc không tìm thấy lỗi cũng có thể là do bộ trường hợp kiểm thử được tạo ra chỉ nhằm kiểm tra những tính năng được làm đúng theo yêu cầu thay vì nhằm tìm kiếm lỗi mới.

1.6 Quy trình kiểm thử phần mềm

Tùy vào từng tổ chức, hệ thống, ngữ cảnh, mức độ rủi ro của phần mềm mà quy trình kiểm thử phần mềm có thể gồm nhiều bước khác nhau. Nhưng nhìn chung mọi quy trình kiểm thử đều có những bước cơ bản (Hình 1-7) dưới đây:



Hình 1-6. Quy trình kiểm thử phần mềm

Lập kế hoạch kiểm thử: Nhiệm vụ quan trọng trong phần lập kế hoạch kiểm thử là xác định được các yếu tố sau:

- Các giai đoạn kiểm thử áp dụng cho dự án
- Các phương pháp kiểm thử
- Các công cụ kiểm thử
- Nguồn lực kiểm thử
- Tài nguyên môi trường kiểm thử, bao gồm các tài nguyên phần cứng và phần mềm
- Mốc bàn giao các tài liệu kiểm thử

Chuẩn bị kiểm thử: Nhiệm vụ chiến lược của giai đoạn này là:

- Tìm hiểu nghiệp vụ của hệ thống phải kiểm thử.
- Xây dựng kịch bản kiểm thử, phát triển các thủ tục và các kịch bản kiểm thử tự động (trong trường hợp kiểm thử tự động).
- Chuẩn bị dữ liệu kiểm thử.
- Xem xét phê duyệt các tài liệu kiểm thử.

Thực thi kiểm thử:

- Thực hiện kiểm thử dựa trên các kịch bản kiểm thử,test script, thủ tục, dữ liệu có sẵn từ bước chuẩn bị kiểm thử.
- Tham gia quá trình quản lý lỗi: báo lỗi, sửa lỗi.

Báo cáo và phân tích dữ liệu kiểm thử:

- Báo cáo kiểm thử.
- Phân tích nguyên nhân và đề xuất các hành động khắc phục.

1.7 Các phương pháp phân tích kiểm thử

Có 2 phương pháp phân tích kiểm thử chính là: phân tích tĩnh và phân tích động.

1.7.1 Phân tích tĩnh

Việc phân tích tĩnh được tiến hành dựa trên việc khảo sát các tài liệu được xây dựng trong quá trình phát triển sản phẩm như tài liệu đặc tả, hồ sơ và mã nguồn phần mềm. Công việc này không động đến việc thực thi chương trình mà chỉ duyệt, lý giải về tất cả các hành vi có thể của chương trình khi được thực thi. Tối ưu hóa các chương trình dịch là các ví dụ về phân tích tĩnh [1].

1.7.2 Phân tích động

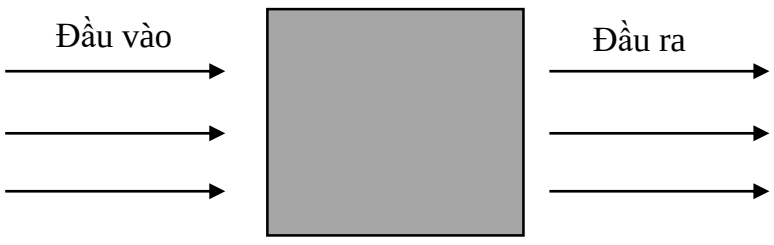
Phân tích động là công việc thực thi chương trình để phát hiện những thất bại có thể có của chương trình, hoặc quan sát cá tính chất nào đó về hành vi và hiệu quả. Vì gần như không thể thực thi chương trình trên tất cả các dữ liệu đầu vào có thể, ta chỉ chọn tập con các dữ liệu đầu vào để thực thi.

1.8 Các kỹ thuật kiểm thử

Ba trong số những kĩ thuật kiểm thử thông dụng nhất bao gồm: Kiểm thử hộp đen, Kiểm thử hộp trắng và Kiểm thử hộp xám.

1.8.1 Kỹ thuật kiểm thử hộp đen

Một trong những kỹ thuật kiểm thử quan trọng trong kiểm thử phần mềm là kiểm thử hộp đen hay còn gọi là kiểm thử hướng dữ liệu, hay kiểm thử hướng vào/ra. Kiểm thử hộp đen xem chương trình như là một "hộp đen". Mục đích là hoàn toàn không quan tâm về cách vận hành và cấu trúc bên trong của chương trình. Thay vào đó, tập trung vào tìm các trường hợp mà chương trình không thực hiện theo các đặc tả của nó. Theo hướng tiếp cận này, dữ liệu kiểm tra được lấy chỉ từ các đặc tả. Kỹ thuật kiểm thử hộp đen được mô tả trong hình 1-8 sau :



Hình 1-7. Kiểm thử hộp đen

a. Mục đích của kiểm thử hộp đen

Kiểm thử hộp đen luôn đóng vai trò quan trọng trong quá trình kiểm thử, mục đích của kiểm thử hộp đen là:

- Chức năng sai hoặc thiếu.
- Lỗi giao diện
- Lỗi trong cấu trúc dữ liệu hoặc truy cập cơ sở dữ liệu từ bên ngoài
- Lỗi hiệu năng
- Lỗi khởi tạo hoặc kết thúc hệ thống

b. Các phương pháp kiểm thử hộp đen

Một số phương pháp tiếp cận chủ yếu dành cho các phương pháp kiểm thử hộp đen bao gồm:

- Phân lớp tương đương – Equivalence partitioning.
- Phân tích giá trị biên – Boundary value analysis.
- Kiểm thử mọi cặp – All-pairs testing.
- Kiểm thử fuzz – Fuzz testing.
- Kiểm thử dựa trên mô hình – Model-based testing.
- Ma trận dấu vết – Traceability matrix.
- Kiểm thử thăm dò – Exploratory testing.
- Kiểm thử dựa trên đặc tả – Specification-base testing.

Kiểm thử hộp đen dựa trên đặc tả tập trung vào kiểm tra tính thiết thực của phần mềm theo những yêu cầu thích hợp. Do đó, kiểm thử viên nhập dữ liệu vào, và chỉ thấy dữ liệu ra từ đối tượng kiểm thử. Mức kiểm thử này thường yêu cầu các ca kiểm thử triệt để được cung cấp cho kiểm thử viên mà khi đó có thể xác minh là đối với dữ liệu đầu vào đã cho, giá trị đầu ra (hay cách thức hoạt động) có giống với giá trị mong muốn đã được xác định trong ca kiểm thử

đó hay không. Kiểm thử dựa trên đặc tả là cần thiết, nhưng không đủ để để ngăn chặn những rủi ro chắc chắn.

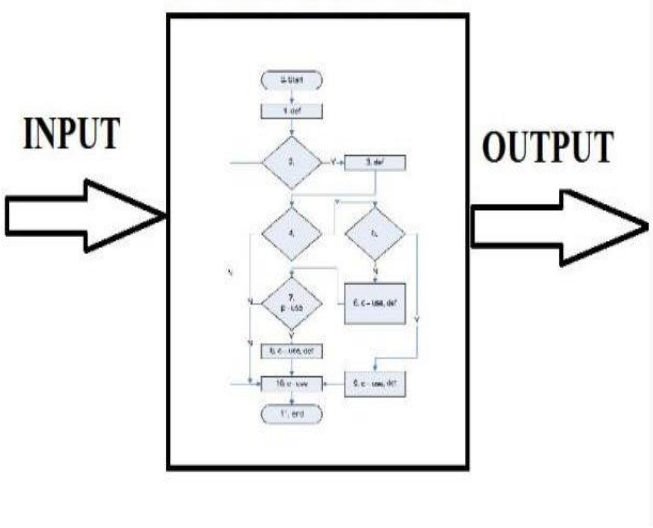
c. Ưu và nhược điểm

Kiểm thử hộp đen không có mối liên quan nào tới mã lệnh, và kiểm thử viên chỉ rất đơn giản tâm niệm là: một mã lệnh phải có lỗi. Sử dụng nguyên tắc " Hãy đòi hỏi và bạn sẽ được nhận", những kiểm thử viên hộp đen tìm ra lỗi mà những lập trình viên đã không tìm ra. Nhưng, mặt khác, người ta cũng nói kiểm thử hộp đen "giống như là đi trong bóng tối mà không có đèn vậy", bởi vì kiểm thử viên không biết các phần mềm được kiểm tra thực sự được xây dựng như thế nào. Đó là lý do mà có nhiều trường hợp mà một kiểm thử viên hộp đen viết rất nhiều ca kiểm thử để kiểm tra một thứ gì đó mà đáng lẽ có thể chỉ cần kiểm tra bằng 1 ca kiểm thử duy nhất, hoặc một số phần của chương trình không được kiểm tra chút nào.

Do vậy, kiểm thử hộp đen có ưu điểm của "một sự đánh giá khách quan", mặt khác nó lại có nhược điểm của "thăm dò mù".

1.8.2 Kỹ thuật kiểm thử hộp trắng

Kỹ thuật kiểm thử hộp trắng hay còn gọi là kiểm thử cấu trúc là một kĩ thuật kiểm thử trái ngược hoàn toàn với kiểm thử hộp đen, kiểm thử hộp trắng hay kiểm thử hướng dựa vào giải thuật cụ thể, vào cấu trúc trúc dữ liệu bên trong của đơn vị phần mềm cần kiểm thử để xác định đơn vị phần mềm đó có thực hiện đúng không. Do đó, kiểm thử viên cần phải có kỹ năng kiến thức nhất định để có thể hiểu chi tiết về đoạn mã cần kiểm thử. Kiểm thử hộp trắng thường tốn rất nhiều thời gian và chi phí nếu mức độ kiểm thử được nâng lên ở cấp kiểm thử tích hợp hay kiểm thử hệ thống.



Hình 1-8. Kiểm thử hộp trắng

a. Các phương pháp kiểm thử hộp trắng

Kiểm thử dòng điều khiển (control flow testing) và kiểm thử dòng dữ liệu (data flow testing) được coi là hai phương pháp tiếp cận chủ yếu trong kĩ thuật kiểm thử hộp trắng.

Kiểm thử dòng điều khiển: là phương pháp tập trung kiểm thử thuật toán chức năng tương ứng với các đường đi (dòng điều khiển) của chương trình. Tuy nhiên chỉ một phương pháp này là chưa đủ để phát hiện tất cả các lỗi tiềm ẩn trong chương trình.

Kiểm thử dòng dữ liệu: là phương pháp cho phép ta phát hiện những lỗi thường xuất hiện tại các biến được sử dụng trong chương trình, đơn vị chương trình.

Bằng cách áp dụng cả hai phương pháp trên sẽ giúp chúng ta đảm bảo về chất lượng của sản phẩm phần mềm tốt hơn.

b. Ưu điểm và nhược điểm

Kiểm thử hộp trắng giúp chúng ta kiểm tra được toàn bộ mã nguồn của chương trình, phát hiện lỗi tại chỗ và có thể áp dụng được các công cụ giúp kiểm thử tự động. Tuy nhiên, kiểm thử hộp trắng yêu cầu kiểm thử viên phải

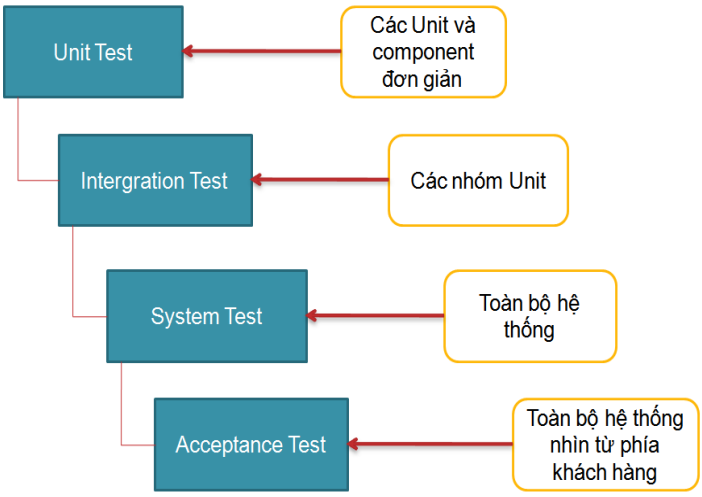
am hiểu về cấu trúc mã lệnh chương trình. Do đó đòi hỏi tài nguyên nhân lực và máy tốn kém. Bên cạnh đó cũng có khả năng tồn tại các tổ hợp lệnh khác nhau gây lỗi, không kiểm thử hết đường đi với các vòng lặp lớn, phức tạp.

1.8.3 Kiểm thử hộp xám

Kiểm thử hộp xám là kỹ thuật kiểm thử có sự kết hợp giữa kiểm thử hộp đen và kiểm thử hộp trắng. Trong kiểm thử hộp đen, kiểm thử viên kiểm thử các hạng mục mà không cần biết cấu trúc bên trong của nó, còn trong kiểm thử hộp trắng thì kiểm thử viên biết được cấu trúc bên trong của chương trình. Trong kiểm thử hộ xám, cấu trúc bên trong sản phẩm chỉ được biết một phần, kiểm thử viên truy cập vào cấu trúc dữ liệu bên trong và thuật toán của chương trình với mục đích là thiết kế ca kiểm thử, nhưng khi kiểm thử thì chỉ kiểm thử như là người dùng cuối hoặc là ở mức hộp đen.

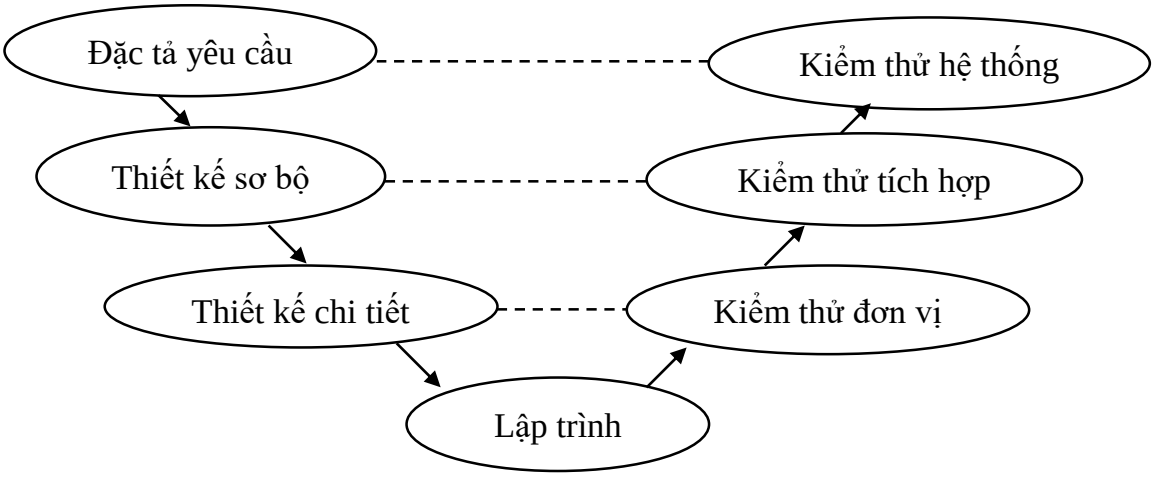
1.9 Các cấp độ kiểm thử

Kiểm thử phần mềm gồm có các cấp độ: Kiểm thử đơn vị, Kiểm thử tích hợp, Kiểm thử hệ thống và Kiểm thử chấp nhận sản phẩm.



Hình 1-9. Các cấp độ kiểm thử

Để thấy được rõ hơn về các cấp độ kiểm thử trong từng giai đoạn phát triển phần mềm, các cấp độ kiểm thử phần mềm thể hiện trong mô hình phát triển thác nước của vòng đời phát triển phần mềm dưới đây:



Hình 1-10. Kiểm thử phần mềm trong mô hình thác nước trừu tượng

1.9.1 Kiểm thử đơn vị

Một đơn vị chương trình là một thành phần phần mềm nhỏ nhất mà ta có thể kiểm thử được. Ví dụ, các hàm *(Function)*, thủ tục *(Procedure)*, lớp *(Class)* hay phương thức *(Method)* đều có thể được xem là đơn vị.

Mỗi đơn vị chương trình được chọn để kiểm tra thường có kích thước nhỏ và chức năng hoạt động đơn giản, chúng ta không khó khăn gì trong việc tổ chức kiểm thử, ghi nhận và phân tích kết quả kiểm thử. Nếu phát hiện lỗi, việc xác định nguyên nhân và khắc phục cũng tương đối dễ dàng vì chỉ khoanh vùng trong một đơn vị đang kiểm tra. Một nguyên lý đúc kết từ thực tiễn: thời gian tốn cho kiểm thử đơn vị sẽ được đền bù bằng việc tiết kiệm rất nhiều thời gian và chi phí cho việc kiểm thử và sửa lỗi ở các mức kiểm thử sau đó.

Kiểm thử đơn vị thường do lập trình viên thực hiện. Công đoạn này cần được thực hiện càng sớm càng tốt trong giai đoạn lập trình và xuyên suốt chu kỳ phát triển phần mềm. Thông thường, kiểm thử đơn vị đòi hỏi kiểm thử viên có kiến thức về thiết kế và mã nguồn của chương trình. Mục đích của kiểm thử đơn vị là bảo đảm thông tin được xử lý và xuất (khỏi đơn vị) là chính xác, trong mối tương quan với dữ liệu nhập và chức năng của đơn vị. Điều này thường đòi hỏi tất cả các nhánh bên trong đơn vị đều phải được kiểm tra để phát hiện nhánh phát sinh lỗi. Một nhánh thường là một chuỗi các lệnh được thực thi trong một

đơn vị. Ví dụ: chuỗi các lệnh sau điều kiện If và nằm giữa then ... else là một nhánh. Thực tế việc chọn lựa các nhánh để đơn giản hóa việc kiểm thử và quét hết đơn vị đòi hỏi phải có kỹ thuật, đôi khi phải dùng thuật toán để chọn lựa.

Cùng với các mục kiểm thử khác, kiểm thử đơn vị cũng đòi hỏi phải chuẩn bị trước các ca kiểm thử (*Test case*) hoặc kịch bản kiểm thử (*Test script*), trong đó chỉ định rõ dữ liệu đầu vào, các bước thực hiện và dữ liệu đầu ra mong muốn. Các ca kiểm thử và kịch bản kiểm thử này nên được giữ lại để tái sử dụng.

1.9.2 Kiểm thử tích hợp

Kiểm thử tích hợp kết hợp các thành phần của một ứng dụng và kiểm thử như một ứng dụng đã hoàn thành. Trong khi kiểm thử đơn vị kiểm tra các thành phần và đơn vị riêng lẻ thì kiểm thử tích hợp kết hợp chúng lại với nhau và kiểm tra sự giao tiếp giữa chúng.

Hai mục tiêu chính của kiểm thử tích hợp:

- Phát hiện lỗi giao tiếp xảy ra giữa các đơn vị.
- Tích hợp các đơn vị đơn lẻ thành các hệ thống nhỏ (*Subsystem*) và cuối cùng là nguyên hệ thống hoàn chỉnh (*System*) chuẩn bị cho kiểm thử ở mức hệ thống (*System Test*).

Trong kiểm thử đơn vị, lập trình viên cố gắng phát hiện lỗi liên quan đến chức năng và cấu trúc nội tại của đơn vị. Có một số phép kiểm thử đơn giản trên giao tiếp giữa đơn vị với các thành phần liên quan khác, tuy nhiên mọi giao tiếp liên quan đến đơn vị chỉ thật sự được kiểm tra đầy đủ khi các đơn vị tích hợp với nhau trong khi thực hiện kiểm thử tích hợp.

Trừ một số ít ngoại lệ, kiểm thử tích hợp chỉ nên thực hiện trên những đơn vị đã được kiểm tra cẩn thận trước đó bằng kiểm thử đơn vị, và tất cả các lỗi mức đơn vị đã được sửa chữa. Một số người hiểu sai rằng đơn vị một khi đã

qua giai đoạn kiểm thử đơn vị với các giao tiếp giả lập thì không cần phải thực kiểm thử tích hợp nữa. Thực tế việc tích hợp giữa các đơn vị dẫn đến những tình huống hoàn toàn khác.

Một chiến lược cần quan tâm trong kiểm thử tích hợp là nên tích hợp dần từng đơn vị. Một đơn vị tại một thời điểm được tích hợp vào một nhóm các đơn vị khác đã tích hợp trước đó và đã hoàn tất các đợt kiểm thử tích hợp trước đó. Lúc này, ta chỉ cần kiểm thử giao tiếp của đơn vị mới thêm vào với hệ thống các đơn vị đã tích hợp trước đó, điều này sẽ làm cho số lượng ca kiểm thử giảm đi rất nhiều, và sai sót sẽ giảm đáng kể.

Có 4 loại kiểm thử trong kiểm thử tích hợp:

- ***Kiểm thử cấu trúc (Structure Test):*** Tương tự kiểm thử hộp trắng, kiểm thử cấu trúc nhằm bảo đảm các thành phần bên trong của một chương trình chạy đúng và chú trọng đến hoạt động của các thành phần cấu trúc nội tại của chương trình chẳng hạn các câu lệnh và nhánh bên trong.
- ***Kiểm thử chức năng (Functional Test):*** Tương tự kiểm thử hộp đen, kiểm thử chức năng chỉ chú trọng đến chức năng của chương trình, mà không quan tâm đến cấu trúc bên trong, chỉ khảo sát chức năng của chương trình theo yêu cầu kỹ thuật.
- ***Kiểm thử hiệu năng (Performance Test):*** Kiểm thử việc vận hành của hệ thống.
- ***Kiểm thử khả năng chịu tải (Stress Test):*** Kiểm thử các giới hạn của hệ thống.

1.9.3 Kiểm thử hệ thống

Mục đích kiểm thử hệ thống là kiểm thử thiết kế và toàn bộ hệ thống (sau khi tích hợp) có thỏa mãn yêu cầu đặt ra hay không.

Kiểm thử hệ thống bắt đầu khi tất cả các bộ phận của phần mềm đã được tích hợp thành công. Thông thường loại kiểm thử này tốn rất nhiều công sức và thời gian. Trong nhiều trường hợp, việc kiểm thử đòi hỏi một số thiết bị phụ trợ, phần mềm hoặc phần cứng đặc thù, đặc biệt là các ứng dụng thời gian thực, hệ thống phân bố, hoặc hệ thống nhúng. Ở mức độ hệ thống, người kiểm thử cũng tìm kiếm các lỗi, nhưng trọng tâm là đánh giá về hoạt động, thao tác, sự tin cậy và các yêu cầu khác liên quan đến chất lượng của toàn hệ thống.

Điểm khác nhau then chốt giữa kiểm thử tích hợp và kiểm thử hệ thống là kiểm thử hệ thống chú trọng các hành vi và lỗi trên toàn hệ thống, còn kiểm thử tích hợp chú trọng sự giao tiếp giữa các đơn thể hoặc đối tượng khi chúng làm việc cùng nhau. Thông thường ta phải thực hiện kiểm thử đơn vị và kiểm thử tích hợp để bảo đảm mọi Unit và sự tương tác giữa chúng hoạt động chính xác trước khi thực hiện kiểm thử hệ thống.

Sau khi hoàn thành kiểm thử tích hợp, một hệ thống phần mềm đã được hình thành cùng với các thành phần đã được kiểm tra đầy đủ. Tại thời điểm này, lập trình viên hoặc kiểm thử viên bắt đầu kiểm thử phần mềm như một hệ thống hoàn chỉnh. Việc lập kế hoạch cho kiểm thử hệ thống nên bắt đầu từ giai đoạn hình thành và phân tích các yêu cầu.

Kiểm thử hệ thống kiểm thử cả các hành vi chức năng của phần mềm lẫn các yêu cầu về chất lượng như độ tin cậy, tính tiện lợi khi sử dụng, hiệu năng và bảo mật. Mức kiểm thử này đặc biệt thích hợp cho việc phát hiện lỗi giao tiếp với phần mềm hoặc phần cứng bên ngoài, chẳng hạn các lỗi "tắc nghẽn" (deadlock) hoặc chiếm dụng bộ nhớ. Sau giai đoạn kiểm thử hệ thống, phần mềm thường đã sẵn sàng cho khách hàng hoặc người dùng cuối cùng kiểm thử chấp nhận sản phẩm (*Acceptance Test*) hoặc dùng thử (*Alpha/Beta Test*).

Đòi hỏi nhiều công sức, thời gian và tính chính xác, khách quan kiểm thử hệ thống thường được thực hiện bởi một nhóm kiểm thử viên hoàn toàn độc lập

với nhóm phát triển dự án. Bản thân kiểm thử hệ thống lại gồm nhiều loại kiểm thử khác nhau, phổ biến nhất gồm:

- ***Kiểm thử chức năng (Functional Test):*** Bảo đảm các hành vi của hệ thống thỏa mãn đúng yêu cầu thiết kế.
- ***Kiểm thử hiệu năng (Performance Test):*** Bảo đảm tối ưu việc phân bổ tài nguyên hệ thống (ví dụ bộ nhớ) nhằm đạt các chỉ tiêu như thời gian xử lý hay đáp ứng câu truy vấn, v.v.
- ***Kiểm thử bảo mật (Security Test):*** Bảo đảm tính toàn vẹn, bảo mật của dữ liệu và của hệ thống.

a. Kiểm thử chức năng

Việc kiểm thử chức năng chú trọng đến hai phần chính là kiểm thử giao diện người dùng (User interface) và kiểm thử luồng nghiệp vụ (Bussiness Flow Testing).

Kiểm thử giao diện người dùng: là việc kiểm tra các tương tác của người dùng với phần mềm. Mục tiêu của kiểm thử giao diện là để đảm bảo rằng giao diện người dùng cung cấp cho người sử dụng cách truy cập và sử dụng các chức năng của hệ thống một cách thích hợp. Ngoài ra, kiểm thử giao diện còn để đảm bảo rằng các đối tượng trên giao diện giống như thiết kế và phù hợp với tổ chức hoặc chuyên ngành.

Mục đích kiểm thử	Kiểm tra việc sử dụng thông qua mục tiêu kiểm thử phản ánh đúng các chức năng và yêu cầu nghiệp vụ, bao gồm màn hình đến màn hình, trường đến trường và sử dụng các phương pháp truy cập(phím tabs, di chuột, tổ hợp phím).

Cách thực hiện	Tạo ra và chỉnh sửa kịch bản kiểm thử cho mỗi màn hình để kiểm tra việc sử dụng đúng cách và tình trạng các đối tượng cho mỗi màn hình và đối tượng của ứng dụng.
Điều kiện hoàn thành	Mỗi màn hình được kiểm tra thành công đúng với phiên kiểm tra hoặc phạm vi chấp nhận được.
Ghi chú	Không phải toàn bộ các thuộc tính của đối tượng đều truy cập được.

Hình 1-11. Kiểm thử giao diện người dùng

Kiểm thử luồng nghiệp vụ: Mục đích của kiểm thử luồng nghiệp vụ là kiểm tra các yêu cầu chức năng và nghiệp vụ của hệ thống bao gồm các hoạt động để kiểm tra tính đúng đắn của dữ liệu, quy trình, báo cáo và việc thực hiện đúng những quy tắc nghiệp vụ. Kiểu kiểm thử này dựa vào kỹ thuật kiểm thử hộp đen, tức là kiểm tra ứng dụng và các xử lý bên trong ứng dụng bằng cách tương tác với ứng dụng thông qua giao diện người sử dụng và phân tích các kết quả hoặc đầu ra. Bảng sau liệt kê một số gợi ý đối với mỗi ứng dụng:

Mục đích kiểm thử	Đảm bảo mục tiêu kiểm thử đúng đắn của chức năng, bao gồm dữ liệu đầu vào, xử lý dữ liệu và dữ liệu nhận được. Kiểm thử chức năng đảm bảo các yêu cầu sau: Nhập dữ liệu hợp lệ thì chương trình phải cho nhập Luồng nghiệp vụ đúng Quá trình xử lý dữ liệu và kết quả đầu ra phải đúng Phục hồi được dữ liệu

Cách thực hiện	Thực hiện các chức năng, sử dụng các dữ liệu hợp lệ và không hợp lệ để kiểm tra. Cụ thể như sau: 1) Kết quả mong đợi với dữ liệu hợp lệ. 2) Lỗi thích hợp hoặc thông báo hiển thị khi dữ liệu không hợp lệ. 3) Mỗi quy tắc nghiệp vụ đều được áp dụng đúng.
Điều kiện hoàn thành	Toàn bộ kế hoạch kiểm thử đã được thực hiện. Toàn bộ các lỗi phát hiện ra đã được ghi nhận.
Ghi chú	

Hình 1-12. Kiểm thử luồng nghiệp vụ

b. Kiểm thử hiệu năng

Mục đích của kiểm thử hiệu năng là kiểm tra các yêu cầu về hiệu năng có đạt được hay không.

Mục đích kiểm thử	Kiểm tra các biểu hiện về hiệu năng cho các giao dịch hoặc chức năng nghiệp vụ theo những điều kiện sau: Khối lượng công việc bình thường đã biết trước. Khối lượng công việc xấu đã biết trước.
Cách thực hiện	Sử dụng các thủ tục cho kiểm thử luồng nghiệp vụ: Chỉnh sửa file dữ liệu để tăng số lượng các giao dịch hoặc scripts để tăng số tương tác xảy ra trong mỗi giao dịch.

	Scripts phải được chạy trên một máy (trường hợp tốt nhất để đánh giá người dùng đơn lẻ, giao dịch đơn lẻ) và phải lặp lại trên nhiều máy trạm.
Điều kiện hoàn thành	Giao dịch đơn lẻ hoặc người dùng đơn lẻ: Thực hiện thành công kịch bản kiểm thử không có lỗi và trong phạm vi mong đợi hoặc thời gian phản hồi cho mỗi giao dịch. Nhiều giao dịch hoặc nhiều người dùng: Thực hiện thành công test script không có lỗi và trong thời gian chấp nhận được.

Hình 1-13. Kiểm thử hiệu năng

c. Kiểm thử bảo mật

Kiểm thử an toàn thông tin tập trung vào hai lĩnh vực bảo mật chính: Bảo mật ở mức ứng dụng và Bảo mật ở mức hệ thống.

Bảo mật mức ứng dụng đảm bảo rằng, dựa trên bảo mật đã yêu cầu, người dùng bị hạn chế sử dụng một số chức năng hoặc tình huống sử dụng, hoặc bị hạn chế trong giới hạn dữ liệu phù hợp với họ. Ví dụ, người dùng có thể được phép nhập dữ liệu để tạo tài khoản nhưng chỉ có người quản lý có thể xóa chúng. Nếu là bảo mật ở mức dữ liệu, việc kiểm thử đảm bảo rằng "người dùng nhóm 1" có thể nhìn thấy các thông tin khách hàng, bao gồm dữ liệu tài chính, tuy nhiên "người dùng nhóm 2" chỉ nhìn thấy các thông tin chung chung cho cùng một khách hàng.

Bảo mật mức hệ thống đảm bảo rằng chỉ những người dùng được cho quyền truy cập vào hệ thống mới có khả năng truy cập vào ứng dụng và chỉ bằng các cổng thích hợp.Mục đích kiểm thử	Bảo mật mức ứng dụng: Đảm bảo rằng một người dùng chỉ có thể truy cập vào những chức năng hoặc dữ liệu mà nhóm người dùng đó được phép. Bảo mật mức hệ thống: Đảm bảo rằng chỉ những người được phép truy cập hệ thống và ứng dụng được phép truy cập chúng.
Cách thực hiện	- Bảo mật ứng dụng: Xác định và liệt kê từng nhóm người dùng và các chức năng hoặc dữ liệu mà họ được phép truy cập. - Tạo kịch bản kiểm thử cho mỗi nhóm người dùng và kiểm tra từng quyền bằng cách tạo các giao dịch xác định cho mỗi nhóm. - Sửa lại nhóm người dùng và chạy lại tình huống kiểm thử cho cùng những người dùng. Với mỗi trường hợp, kiểm tra các chức năng thêm vào hoặc dữ liệu có đúng không hay bị từ chối. - Truy cập mức hệ thống: tham khảo các điều kiện đặc biệt dưới đây.
Điều kiện hoàn thành	- Với mỗi nhóm người dùng đều có các chức năng hoặc dữ liệu thích hợp, và toàn bộ các chức năng giao dịch đều như dự kiến và chạy trong các kiểm thử chức năng ứng dụng trước đó.

Ghi chú	- Truy cập vào hệ thống phải được xem xét hoặc thảo luận với quản trị hệ thống hoặc quản trị mạng, có thể không cần nếu nó là chức năng của quản trị mạng hoặc quản trị hệ thống.

Hình 1-14. Kiểm thử bảo mật

Nhìn từ quan điểm người dùng, các cấp độ kiểm thử trên rất quan trọng. Chúng bảo đảm hệ thống đủ khả năng làm việc trong môi trường thực.

Lưu ý là không nhất thiết phải thực hiện tất cả các loại kiểm thử nêu trên. Tùy yêu cầu và đặc trưng của từng hệ thống, tùy khả năng và thời gian cho phép của dự án, khi lập kế hoạch, người Quản lý dự án sẽ quyết định áp dụng những loại kiểm thử nào.

1.9.4 Kiểm thử chấp nhận sản phẩm

Thông thường, sau giai đoạn kiểm thử hệ thống là kiểm thử chấp nhận, được khách hàng thực hiện (hoặc ủy quyền cho một nhóm thứ ba thực hiện). Mục đích của kiểm thử chấp nhận là để chứng minh phần mềm thỏa mãn tất cả yêu cầu của khách hàng và khách hàng chấp nhận sản phẩm (và trả tiền thanh toán hợp đồng).

Kiểm thử chấp nhận có ý nghĩa hết sức quan trọng, mặc dù trong hầu hết mọi trường hợp, các phép kiểm thử của kiểm thử hệ thống và kiểm thử chấp nhận gần như tương tự, nhưng bản chất và cách thức thực hiện lại rất khác biệt.

Đối với những sản phẩm dành bán rộng rãi trên thị trường cho nhiều người sử dụng, thông thường sẽ thông qua hai loại kiểm thử gọi là kiểm thử Alpha – ***Alpha Test*** và kiểm thử Beta – ***Beta Test***. Với kiểm thử Alpha, người dùng kiểm thử phần mềm ngay tại nơi phát triển phần mềm, lập trình viên sẽ ghi nhận các lỗi hoặc phản hồi, và lên kế hoạch sửa chữa. Với kiểm thử Beta, phần mềm

sẽ được gửi tới cho người dùng để kiểm thử ngay trong môi trường thực, lỗi hoặc phản hồi cũng sẽ gửi ngược lại cho lập trình viên để sửa chữa.

Thực tế cho thấy, nếu khách hàng không quan tâm và không tham gia vào quá trình phát triển phần mềm thì kết quả kiểm thử chấp nhận sẽ sai lệch rất lớn, mặc dù phần mềm đã trải qua tất cả các kiểm thử trước đó. Sự sai lệch này liên quan đến việc hiểu sai yêu cầu cũng như sự mong chờ của khách hàng. Ví dụ đôi khi một phần mềm xuất sắc vượt qua các phép kiểm thử về chức năng thực hiện bởi nhóm thực hiện dự án, nhưng khách hàng khi kiểm thử sau cùng vẫn thất vọng vì bố cục màn hình nghèo nàn, thao tác không tự nhiên, không theo tập quán sử dụng của khách hàng v.v...

Gắn liền với giai đoạn kiểm thử chấp nhận thường là một nhóm những dịch vụ và tài liệu đi kèm, phổ biến như hướng dẫn cài đặt, sử dụng v.v... Tất cả tài liệu đi kèm phải được cập nhật và kiểm thử chặt chẽ.

1.9.5 Một số cấp độ kiểm thử khác

Ngoài các cấp độ trên, còn một số cấp độ kiểm thử khác như:

Kiểm thử hồi quy – Regression Testing: là "Sự kiểm tra lại có lựa chọn của một hệ thống hay thành phần để xác minh là những sự thay đổi không gây ra những hậu quả không mong muốn,v.v". Trên thực tế, quan niệm này là chỉ ra rằng phần mềm mà đã qua được các kiểm tra trước đó vẫn có thể được kiểm tra lại. *Theo Beizer* định nghĩa đó là sự lặp lại các kiểm tra để chỉ ra rằng cách hoạt động của phần mềm không bị thay đổi, ngoại trừ tới mức như yêu cầu. Hiển nhiên là sự thỏa hiệp phải được thực hiện giữa sự đảm bảo được đưa ra bởi kiểm thử hồi quy mỗi lần thực hiện một sự thay đổi và những tài nguyên được yêu cầu thực hiện điều đó.

Kiểm thử tính đúng – Correctness Testing: Tính đúng đắn là yêu cầu tối thiểu của phần mềm, là mục đích chủ yếu của kiểm thử. Kiểm thử tính đúng sẽ cần một kiểu người đáng tin nào đó, để chỉ ra cách hoạt động đúng đắn từ

cách hoạt động sai lầm. Kiểm thử viên có thể biết hoặc không biết các chi tiết bên trong của các modun phần mềm được kiểm thử, ví dụ luồng điều khiển, luồng dữ liệu, v.v. Do đó, hoặc là quan điểm hộp trắng, hoặc là quan điểm hộp đen có thể được thực hiện trong kiểm thử phần mềm.

1.10 Kỹ thuật xác định các yếu tố trong ca kiểm thử

1.10.1 Ca kiểm thử

Ca kiểm thử là tập các điều kiện đầu vào, điều kiện thực hiện và kết quả mong đợi được phát triển, được viết cho một mục đích nào đó nhằm xác định sự tuân thủ đối với đặc tả yêu cầu phần mềm của chương trình. Một ca kiểm thử bao gồm các thành phần trong hình 1-16 sau:

Test Case ID	**Test Case Description**	**Pre-Requisites**	**Steps**	**Execution Steps**	**Expected Results**	**Actual Results**	**Result**	**Note**
01								
…								
n								

Hình 1-15. Mẫu ca kiểm thử

Trong đó:

- Test Case ID: *là ô đánh số thứ tự của ca kiểm thử nhằm mục đích lần lượt, tránh bỏ sót ca kiểm thử.*
- Test Case Description: *là ô miêu tả nội dung của ca kiểm thử sắp tiến hành.*
- Pre-Requisites: *là ô miêu tả diễn các tiền điều kiện cần thiết để tiến hành ca kiểm thử.*

- Steps: *là số đánh số thứ tự các bước thực hiện ca kiểm thử, để sau khi kiểm thử có thể tìm thấy lỗi chi tiết hơn ở từng mục.*
- Execution Steps: *là ô miêu tả chi tiết từng hành động thực hiện ca kiểm thử của ca kiểm thử.*
- Expected Results: *là ô thể hiện các kết quả mong muốn đầu ra, như là output của các dữ liệu đầu vào.*
- Actual Results: *là ô phản ánh kết quả đúng sau khi thực hiện từng bước của ca kiểm thử.*
- Result: *kết quả tổng quan của ca kiểm thử, thường biểu diễn là Pass/Fail, thành công hoặc thất bại so với mong muốn đầu ra của ca kiểm thử.*
- Note: *là ô để kiểm thử viên ghi chú những thông tin cần lưu ý trong quá trình thực hiện ca kiểm thử.*

1.10.2 Một số kỹ thuật xác định ca kiểm thử

a. Kỹ thuật phần vùng tương đương

Định nghĩa: Phân vùng tương đương là phương pháp kiểm thử hộp đen chia miền đầu vào của một chương trình thành các lớp dữ liệu, từ đó suy dẫn ra các ca kiểm thử. Thiết kế ca kiểm thử cho phân vùng tương đương dựa trên sự đánh giá về các vùng tương đương với một điều kiện vào. Vùng tương đương biểu thị một tập cho các trạng thái hợp lệ hay không hợp lệ đối với điều kiện vào. Thiết kế ca kiểm thử bằng phân vùng tương đương tiến hành theo hai bước:

Bước 1: Xác định các lớp tương đương.

Bước 2: Xác định các ca kiểm thử.

Xác định các lớp tương đương: Có hai kiểu lớp tương đương được xác định: lớp tương đương hợp lệ mô tả các đầu vào và lớp tương đương không hợp lệ mô tả. Để xác định các lớp tương đương được chính xác có thể áp dụng các nguyên tắc sau đây.

Quy tắc 1: Nếu trạng thái đầu vào định rõ giới hạn của các giá trị, thì xác định một lớp tương đương hợp lệ và hai lớp tương đương không hợp lệ.

Quy tắc 2: Nếu một trạng thái đầu vào xác định số giá trị, xác định một lớp tương đương hợp lệ và hai lớp tương đương bất hợp lệ.

Quy tắc 3: Nếu 1 trạng thái đầu vào chỉ định tập các giá trị đầu vào và chương trình sử dụng mỗi giá trị là khác nhau, xác định một lớp tương đương hợp lệ cho mỗi loại và một lớp tương đương không hợp lệ.

Quy tắc 4: Nếu 1 trạng thái đầu vào chỉ định một tình huống "chắc chắn – must be", xác định một lớp tương đương hợp lệ và một lớp tương đương không hợp lệ.

Xác định các ca kiểm thử: Với các lớp tương đương xác định được ở bước trên, bước thứ hai là sử dụng các lớp tương đương đó để xác định các ca kiểm thử. Quá trình này như sau:

1. Gán 1 số duy nhất cho mỗi lớp tương đương.

2. Cho đến khi tất cả các lớp tương đương hợp lệ được bao phủ bởi (hợp nhất thành) các ca kiểm thử, viết 1 ca kiểm thử mới bao phủ càng nhiều các lớp tương đương đó chưa được bao phủ càng tốt.

3. Cho đến khi các ca kiểm thử của bạn đã bao phủ tất cả các lớp tương đương không hợp lệ, viết một ca kiểm thử mà bao phủ một và chỉ một trong các lớp tương đương không hợp lệ chưa được bao phủ.

4. Lý do mà mỗi ca kiểm thử riêng bao phủ các trường hợp không hợp lệ là vì các kiểm tra đầu vào không đúng nào đó che giấu hoặc thay thế các kiểm tra đầu vào không đúng khác.

Ưu điểm: Vì mỗi vùng tương đương ta chỉ cần kiểm thử trên các phần tử đại diện nên số lượng ca kiểm thử được giảm đi khá nhiều nhờ đó mà thời gian thực hiện kiểm thử cũng giảm đáng kể.

Nhược điểm: Không phải với bất kỳ bài toán nào đều có thể áp dụng kỹ thuật này. Có thể bị thiếu lỗi ở biên nếu chỉ chọn giá trị ở khoảng giữa của miền tương đương. Vì vậy khi phần lớn các lỗi được tìm thấy lúc kiểm tra giá trị ở biên của các phân vùng thì chúng ta nên tìm hiểu thêm một kỹ thuật nữa là *Phân tích giá trị biên.*

b. Phân tích giá trị biên

Đây là một trong những kỹ thuật kiểm thử phần mềm, trong đó các ca kiểm thử được thiết kế bao gồm các giá trị tại các biên. Nếu dữ liệu đầu vào được sử dụng là trong giới hạn giá trị biên, nó được cho là Positive testing. Nếu dữ liệu đầu vào được sử dụng là ngoài giới hạn giá trị biên, nó được cho là Negative testing. Mục tiêu là lựa chọn các ca kiểm thử để thực thi giá trị biên.

Phân tích giá trị biên yêu cầu kiểm thử viên có chuyên môn sâu và kinh nghiệm cao. Tuy nhiên, có một số quy tắc chung sau:

1. Nếu một trạng thái đầu vào định rõ giới hạn của các giá trị, hãy viết các ca kiểm thử cho các giá trị cuối của giới hạn, và các ca kiểm thử đầu vào không hợp lệ cho các trường hợp vừa ra ngoài phạm vi.

2. Nếu một trạng thái đầu vào định rõ số lượng giá trị, hãy viết các ca kiểm thử cho con số lớn nhất và nhỏ nhất của các giá trị và một giá trị trên, một giá trị dưới những giá trị này.

3. Sử dụng quy tắc số 1 cho mỗi trạng thái đầu vào.

4. Sử dụng nguyên tắc số 2 cho mỗi trạng thái đầu ra.

5. Nếu đầu vào hay đầu ra của một chương trình là tập được sắp thứ tự (ví dụ, một file tuần tự hay một danh sách định tuyến hay một bảng) tập trung chú ý vào các phần tử đầu tiên và cuối cùng của tập hợp.

6. Sử dụng sự khéo léo của bạn để tìm các điều kiện biên.

Các ca kiểm thử chuẩn được lựa chọn dựa vào quy tắc sau:

- Giá trị biên nhỏ nhất – 1
- Giá trị biên nhỏ nhất
- Giá trị biên lớn nhất
- Giá trị biên lớn nhất + 1

Nhưng nếu bạn muốn kiểm tra sâu hơn thì bạn cũng có thể lựa chọn theo quy tắc:

- Giá trị biên nhỏ nhất – 1
- Giá trị biên nhỏ nhất
- Giá trị biên nhỏ nhất + 1
- Giá trị biên lớn nhất – 1
- Giá trị biên lớn nhất
- Giá trị biên lớn nhất + 1

Phân tích các giá trị biên là phương pháp thiết kế ca kiểm thử bổ sung thêm cho phân lớp tương đương, nhưng khác với phân lớp tương đương ở hai khía cạnh:

1. Phân tích giá trị biên không lựa chọn phần tử bất kỳ nào trong một lớp tương đương là điển hình, mà nó yêu cầu là một hay nhiều phần tử được lựa chọn như vậy mà mỗi cạnh của lớp tương đương đó chính là đối tượng kiểm tra.

2. Ngoài việc chỉ tập trung chú ý vào các trạng thái đầu vào (không gian đầu vào), các ca kiểm thử cũng nhận được bằng việc xem xét không gian kết quả (các lớp tương đương đầu ra).

c. Bảng quyết định

Kĩ thuật bảng quyết định tập trung vào tính lôgíc và các ràng buộc của hệ thống. Bảng quyết định là cách tốt nhất để đối ứng với sự kết hợp của các điều kiện. Lý do cho điều này:

- Bảng quyết định cung cấp một cách có hệ thống các quy tắc kinh doanh phức tạp, rất hữu ích cho cả lập trình viên và kiểm thử viên.
- Bảng quyết định có thể được sử dụng trong thiết kế ca kiểm thử, vì chúng giúp kiểm thử viên tìm được những tác động khi kết hợp các yếu tốt đầu vào khác nhau và các trạng thái phần mềm mà phải thực hiện đúng các quy tắc nghiệp vụ khác.
- Kiểm thử kết hợp có thể là một thách thức vì số lượng các kết hợp điều kiện có thể là rất lớn. Kiểm tra tất cả các kết hợp điều kiện có thể là không thực tế nếu không phải là không thể. Chúng ta phải hài lòng với việc chỉ kiểm thử một phần nhỏ trong số tất cả các kết hợp điều kiện, nhưng lựa chọn kết hợp điều kiện nào để kiểm thử và bỏ qua cũng là vấn đề rất quan trọng. Nếu như không có các kết hợp điều kiện được lựa chọn một cách có hệ thống, một tập tùy ý được sử dụng thì bảng quyết định cũng có thể dẫn đến kiểm thử không hiệu quả. Bảng quyết định có dạng mẫu trong hình 1-17 sau:

		Các ca kiểm thử							
Nguyên nhân	Giá trị	1	2	3	4	5	6	7	8
Nguyên nhân 1	(Y/N)	Y	Y	Y	Y	N	N	N	N
Nguyên nhân 2	(Y/N)	Y	Y	N	N	Y	Y	N	N
Nguyễn nhân 3	(Y/N)	Y	N	Y	N	Y	N	Y	N
Kết quả mong đợi									
Kết quả 1		X			X				X
Kết quả 2			X					X	X

Hình 1-16. Mẫu bảng quyết định

Các bước tạo "Bảng quyết định":

- Liệt kê tất cả các nguyên nhân trong bảng quyết định.
- Tính tổng số lượng kết hợp giữa các nguyên nhân.
- Điền vào các cột với tất cả kết hợp có thể có.
- Rút bớt số lượng các phép kết hợp thừa.
- Kiểm tra các phép kết hợp có bao phủ hết mọi trường hợp không.
- Bổ sung kết quả vào bảng quyết định.

Để xác định số ca kiểm thử xảy, Bảng quyết định dựa vào nguyên nhân và số lượng giá trị lựa chọn đầu vào để tính toán:

=[số lượng giá trị của nguyên nhân]*…*[Số lượng giá trị nguyên nhân n]

Ưu điểm của kỹ thuật này là chúng ta có thể kiểm tra sự kết hợp của các điều kiện mà có thể đã bị thiếu sót và không được thử nghiệm đồng thời có thể tìm thấy khiếm khuyết. Tuy nhiên nếu có quá nhiều kết hợp các điều kiện, sử dụng kỹ thuật này có thể không khả quan hoặc không hợp lý để kiểm tra từng kết hợp điều kiện. Đừng cho rằng tất cả các kết hợp cần phải được kiểm tra, nên ưu tiên kiểm tra những kết hợp quan trọng nhất. Có đầy đủ các kết hợp điều kiện sẽ giúp chúng ta quyết định được kết hợp điều kiện nào cần kiểm tra và chưa cần kiểm tra ở thời điểm nhất định nào đó.

CHƯƠNG 2: KỸ THUẬT TẠO CA KIỂM THỬ TỪ BIỂU ĐỒ LUỒNG DỮ LIỆU

Trong chương này, khóa luận sẽ trình bày tổng quan cơ bản về biểu đồ luồng dữ liệu và kỹ thuật xây dựng ca kiểm thử từ biểu đồ luồng dữ liệu.

2.1 Biểu đồ luồng dữ liệu

Biểu đồ luồng dữ liệu là kết quả của quá trình phân tích thiết kế hệ thống, một loại biểu đồ nhằm mục đích mô tả tập hợp các chức năng xử lý thông tin của hệ thống trong mối quan hệ, mối quan hệ của chúng được thể hiện qua:

- Trình tự trước sau của các chức năng trong tiến trình xử lý.
- Sự kế thừa thông tin giữa các chức năng, nghĩa là thông tin của chức năng này có thể là thông tin đầu vào của chức năng khác.

Biểu đồ luồng dữ liệu là công cụ chính của quá trình phân tích, nhằm mục đích thiết kế trao đổi và tạo lập dữ liệu. Nó thể hiện rõ ràng và khá đầy đủ các nét đặc trưng của hệ thống trong các bước phân tích, thiết kế và trao đổi dữ liệu. Bên cạnh đó, biểu đồ luồng dữ liệu cũng hỗ trợ một số tính năng cho chương trình như sau:

- Xác định yêu cầu của người dùng.
- Lập kế hoạch và minh hoạ những phương án cho phân tích viên và người dùng xem xét.
- Trao đổi giữa những phân tích viên và người dùng trong hệ thống.
- Làm tài liệu đặc tả yêu cầu hình thức và đặc tả thiết kế hệ thống.

2.2 Các thành phần của biểu đồ luồng dữ liệu

Biểu đồ luồng dữ liệu gồm các thành phần sau:

2.2.1 Tiến trình

Khái niệm: Tiến trình là các quá trình biến đổi thông tin, tổ chức lại thông tin, bổ sung thông tin hoặc tạo ra thông tin mới để tổ chức thành thông tin đầu ra phục vụ cho hoạt động của hệ thống như lưu vào kho dữ liệu hoặc gửi cho các chức năng khác.

Biểu diễn: Chức năng xử lí được biểu diễn bằng hình chữ nhật bo tròn trong đó có ghi tên của chức năng

Tên chức năng: Tên tiến trình gồm một động từ cộng với bổ ngữ nếu cần, cho phép hiểu một các vắn tắt chức năng làm gì.

Ví dụ: Tiến trình "Mua hàng" trong hệ thống thông tin "Quản Lý Bán Hàng":

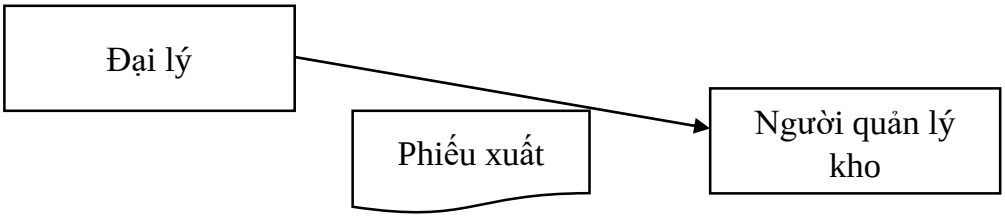
Cần chú ý rằng, tên của chức trong biểu đồ luồng dữ liệu phải trùng với tên đã được đặt trong biểu đồ phân cấp chức năng. Khi xây dựng biểu đồ luồng dữ liệu nếu có chức năng nào không tạo ra thông tin thì nó chưa phải là chức năng trong biểu đồ luồng dữ liệu và khi đó cần phải xem xét lại biểu đồ phân cấp chức năng. Thông thường nên xem xét đến khả năng chức này đã bị tách ra khỏi chức năng khác một cách không hợp lí.

2.2.2 Luồng dữ liệu

Khái niệm: Luồng dữ liệu là luồng thông tin vào hay ra của một chức năng xử lí. Nghĩa là có một thông tin được chuyển đến một chức năng xử lý hoặc chuyển ra khỏi một chức năng như một kết quả xử lý mà không cần quan tâm đến hình thức truyền dẫn (bằng tay, máy fax,v.v.).

Biểu diễn: Luồng dữ liệu trên biểu đồ được biểu diễn bằng mũi tên có hướng trên đó có ghi tên là luồng thông tin mang theo. Mũi tên chỉ hướng của luồng thông tin.

Tên luồng dữ liệu: Tên luồng dữ liệu là tên gồm danh từ cộng với tính từ nếu cần thiết, cho phép hiểu một cách vắn tắt nội dung của dữ liệu được chuyển giao. Các luồng dữ liệu và tên được gán cho chúng là các thông tin "logic" chứ không phải là các tài liệu vật lí – giá mang thông tin. Tuy nhiên trong một số trường hợp trên dòng dữ liệu trùng (hoặc đã quen dùng) với tên tài liệu vật lí. Ví dụ: Một luồng dữ liệu là "Phiếu xuất" đi từ tác nhân trong "Người quản lý kho" đến tác nhân ngoài "Đại lý".



2.2.3 Kho dữ liệu

Khái niệm: Kho dữ liệu là các thông tin cần lưu trữ trong một khoảng thời gian, để sau đó một hay một số chức năng xử lí, hoặc tác nhân trong sử dụng.

Biểu diễn: Kho dữ liệu được biểu diễn bằng cặp đoạn thẳng song song trên đó có ghi tên của kho.

Tên: Kho chứa các dữ liệu nên tên của kho là danh từ kèm theo tính từ nếu cần thiết, nó nói lên nội dung thông tin chứ không phải là giá mang thông tin.

Ví dụ: "Phiếu xuất kho", "Đơn đặt hàng".

D	Phiếu xuất kho		Đơn đặt hàng	D

2.2.4 Tác nhân ngoài

Tác nhân ngoài còn được gọi là *Đối tác*, là một người, nhóm người hay một tổ chức ở bên ngoài lĩnh vực nghiên cứu của hệ thống nhưng có tiếp xúc, trao đổi, thông tin với hệ thống. Sự có mặt của các nhân tố này trên sơ đồ chỉ ra giới hạn của hệ thống, và định rõ mối quan hệ của hệ thống với thế giới bên

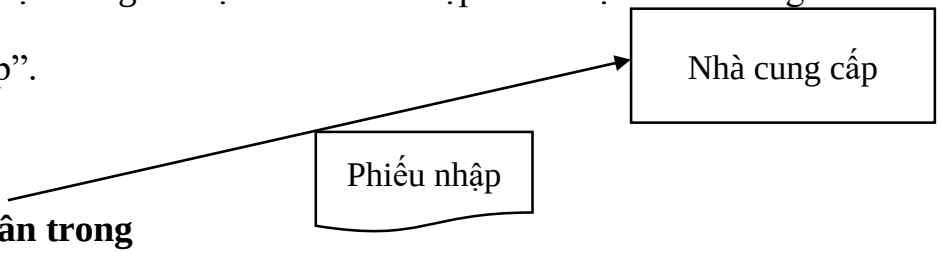
ngoài. Điều đáng chú ý là hiểu nghĩa "ngoài lĩnh vực nghiên cứu" không có nghĩa là bên ngoài tổ chức, chẳng hạn như hệ thống xử lí hàng thì bộ phận kế toán, bộ phận mua hàng và các bộ phận kho vẫn là tác nhân ngoài. Đối với hệ thống tuyển sinh đại học thì tác nhân ngoài vẫn có thể là thí sinh, giáo viên chấm thi và hội đồng tuyển sinh.

Tác nhân ngoài là phần sống còn của hệ thống, chúng là nguồn cung cấp thông tin cho hệ thống cũng như chứng nhận các sản phẩm thông tin từ hệ thống.

Biểu diễn: Bằng hình chữ nhận có tên

Tên: Được xác định bằng các danh từ kèm theo tính từ nếu cần thiết.

Ví dụ: Một luồng dữ liệu là "Phiếu nhập" đến một tác nhân ngoài là "Nhà cung cấp".



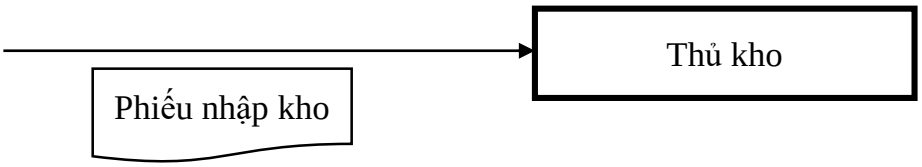
2.2.5 Tác nhân trong

Khái niệm: Tác nhân trong là một chức năng hay một hệ thống con của hệ thống được mô tả ở trong của biểu đồ, nhưng có trao đổi thông tin với các phần tử thuộc trang hiện tại của biểu đồ. Thông thường mọi biểu đồ có thể bao gồm một số trang, đặc biệt là trong các hệ thống phức tạp và với khuôn khổ giấy có hạn thông được truyền giữa các quá trình trên các trang khác nhau được chỉ ra nhờ kí hiệu này, ý nghĩa của tác nhân trong với kí hiệu tương tự như nút tiếp nối của sơ đồ thuật toán.

Biểu diễn: Tác nhận trong biểu diễn bằng hình chữ nhật hở một phía và trong có ghi tên.

Tên tác nhân trong: Được biểu diễn bằng động từ kèm bổ ngữ nếu cần.

Ví dụ: Một luồng dữ liệu là"Phiếu xuất/nhập" đến một tác nhân trong là "Thủ kho".



2.3 Cơ sở sinh ra biểu đồ luồng dữ liệu

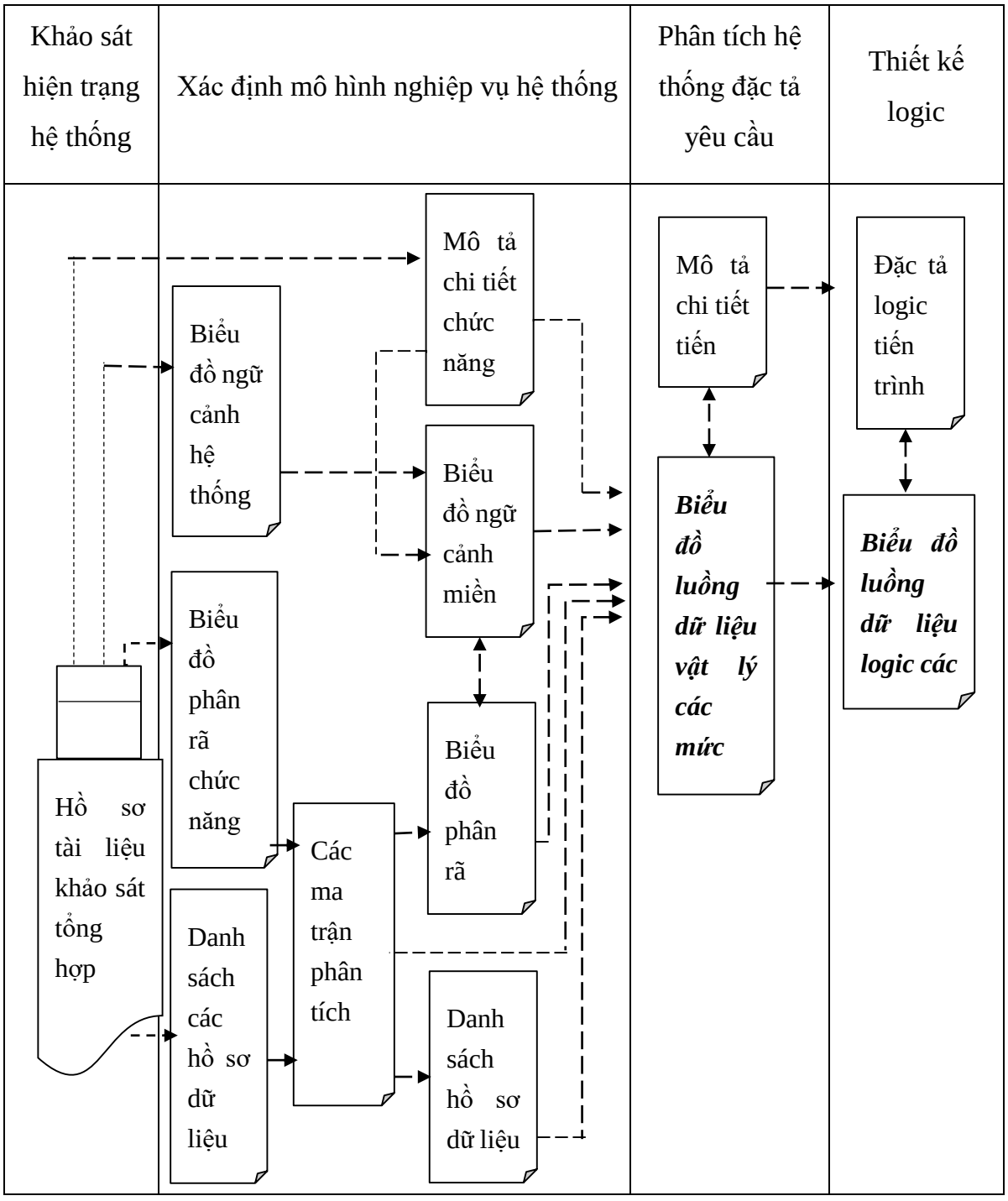
Trước khi tiến hành phân tích biểu đồ luồng dữ liệu, khóa luận trình bày cơ sở để sinh ra biểu đồ luồng dữ liệu.

Biểu đồ luồng dữ liệu là kết quả của công việc phân tích thiết kế hệ thống. Quá trình đó nhằm phục vụ cho việc thiết kế phần mềm một cách hoàn thiện, không thiếu hay thừa chức năng của người dùng. Bên cạnh đó cũng giúp cho kiểm thử viên dùng để hoàn thiện các mức kiểm tra cho phần mềm cho từng giai đoạn phát triển, quá trình hình thành của biểu đồ luồng dữ liệu được thể hiện trong (Hình 2-1). Biểu đồ luồng dữ liệu là một trong những công cụ quan trọng nhất trong việc phân tích hệ thống có cấu trúc. Nó đưa ra được mối liên hệ giữa chức năng hoặc tiến trình của hệ thống với thông tin mà chúng sử dụng. Nó là một phần chủ chốt của đặc tả yêu cầu hệ thống, vì nó xác định thông tin nào phải có mặt trước khi quá trình có thể được tiến hành. Biểu đồ luồng dữ liệu hỗ trợ cho 4 hoạt động chính của nhà phân tích:

1. Phân tích - được dùng để giúp xác định yêu cầu người sử dụng.
2. Thiết kế - dùng để vạch ra kế hoạch và minh hoạ các phương án cho nhà phân tích và người dùng xem xét khi thiết kế hệ thống mới.
3. Liên lạc - giữa nhà phân tích và người sử dụng do tính đơn giản, tính dễ hiểu của nó.
4. Tài liệu - được dùng trong đặc tả yêu cầu hình thức và đặc tả thiết kế hệ thống.

Bên cạnh những ưu điểm thì biểu đồ luồng dữ liệu cũng còn những mặt hạn chế, biểu đồ luồng dữ liệu:

- Không chỉ ra được yếu tố thời gian (VD: Thông tin chuyển từ tiến trình này sang tiến trình khác hết bao nhiêu thời gian).
- Không xác định được trật tự thực hiện các tiến trình (hay các chức năng).
- Không chỉ ra được yếu tố định lượng đối với dữ liệu có liên quan.(tối đa và tối thiểu những thông tin là cơ bản trong quá trình phân tích).

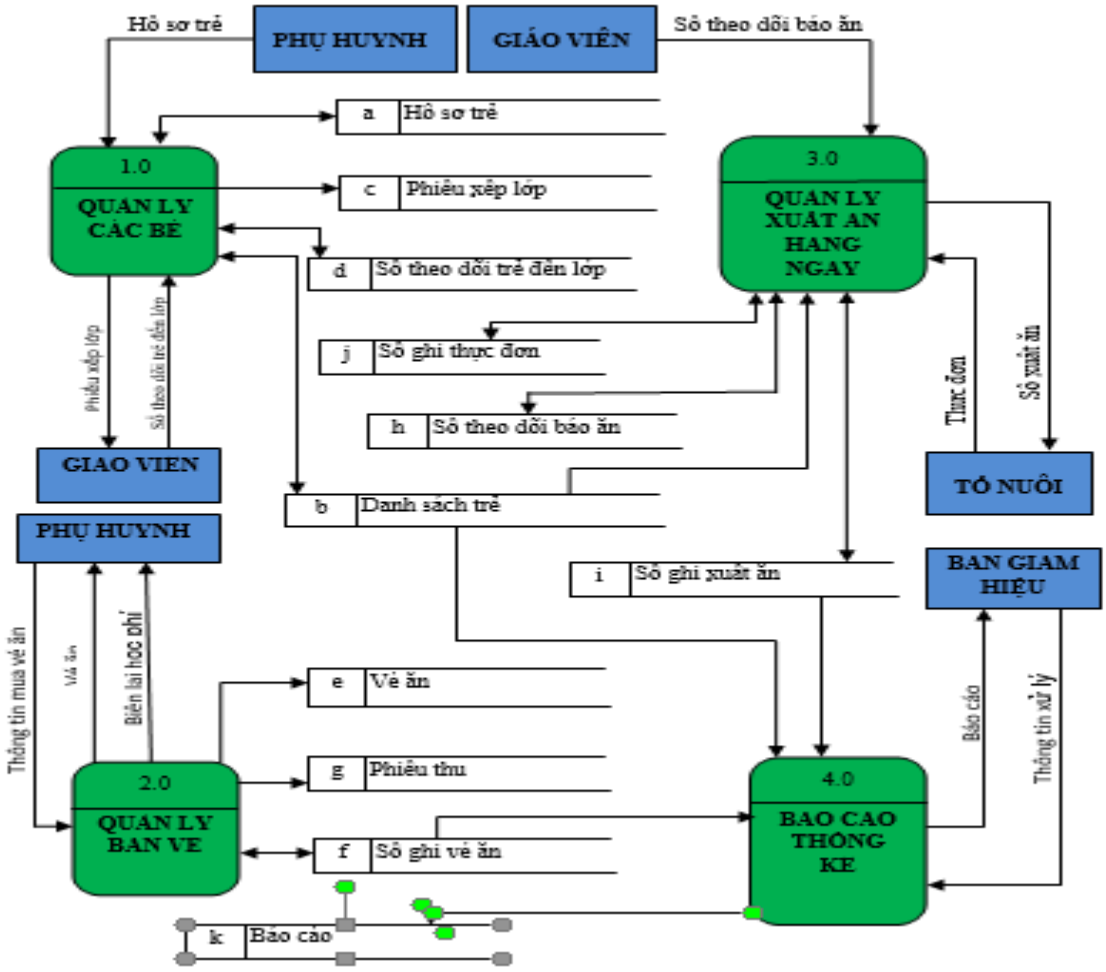


Hình 2-1. Quy trình phát triển biểu đồ luồng dữ liệu

2.4 Phân tích thông tin từ biểu đồ luồng dữ liệu

Trong phần này khóa luận trình bày phương pháp phân tích các thông tin từ biểu đồ luồng dữ liệu để làm cơ sở cho quá trình xây dựng ca kiểm thử. Trong phạm vi phân tích thông tin, em có sử dụng lại biểu đồ luồng dữ liệu đã hoàn thiện của anh cựu sinh viên Đinh Văn Phong khóa luận tốt nghiệp năm 2014. Sau khi tổng hợp và phân tích để ra được biểu đồ luồng dữ liệu

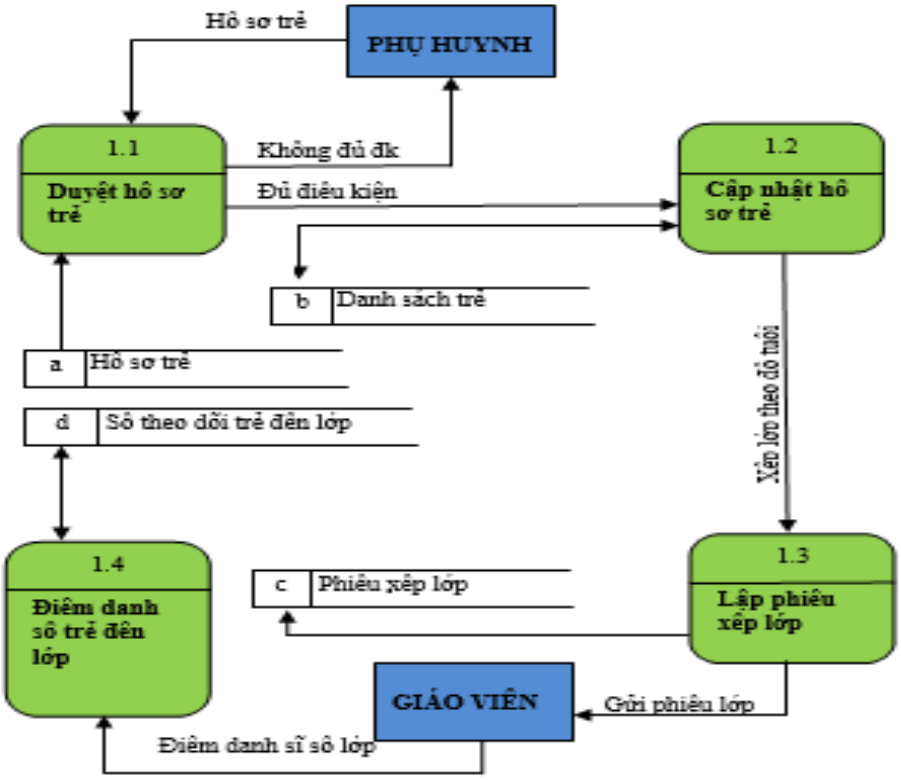
Số đồ luồng dữ liệu mức 0



Hình 2-2. Biểu đồ dữ liệu mức 0

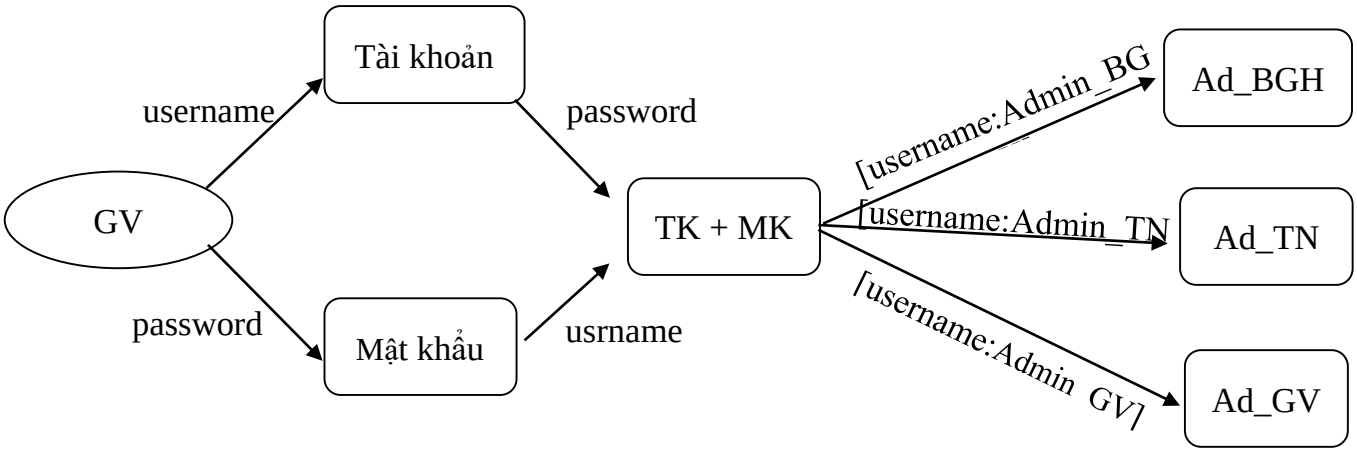
Tiếp tục phân rã để làm mịn dần các chức năng

Sơ đồ luồng dữ liệu mức 1 chức năng Quản lí các học sinh.



Hình 2-3. Biểu đồ luồng dữ liệu mức 1

Tiếp tục quá trình phân rã để làm mịn dần từ biểu đồ luồng dữ liệu mức 1 trên để có được biểu đồ chi tiết các chức năng con, ta có được sơ đồ luồng dữ liệu mức chi tiết 2 chức năng điểm danh.



Từ biểu mức 2 trên, ta có thể rút ra được các yêu cầu của người dùng về chức năng điểm danh trên lớp hàng ngày.

Đặc tả yêu cầu người dùng: Mỗi sáng, giáo viên phụ trách lớp sẽ đăng nhập vào hệ thống, điểm danh sĩ số trên sổ theo dõi trẻ đến lớp của phần mềm, sau đó lưu lại và báo cáo về kế toán tình trạng của các cháu đi học để tổ nuôi lên số lượng suất ăn cho các cháu đi học ngày hôm đó.

Đặc tả yêu cầu hệ thống: Giáo viên được cấp một tài khoản để đăng nhập vào hệ thống dùng để điểm danh danh sách học sinh mỗi lớp hàng ngày. Tài khoản sử dụng là email đã được hệ thống kích hoạt, mật khẩu có độ dài từ 8-32 kí tự, bao gồm kí từ a-z,A-Z và số, không chứa các kí tự đặc biệt @#$%^&*()_.,v.v. Mỗi tài khoản đều được phân quyền khác nhau (BGH, Tổ Nuôi, Giáo Viên).

2.5 Xây dựng ca kiểm thử từ biểu đồ luồng dữ liệu

Từ đặc tả yêu cầu của hệ thống sẽ tiến hành thiết kế các ca kiểm thử cho chức năng của chương trình. Để thiết kế ca kiểm thử chúng ta cần phải xác định được các loại ca kiểm thử phục vụ cho kiểm thử. Thông thường, các kiểm thử viên thiết kế ca kiểm thử ba loại sau:

- Kiểm thử chức năng
- Kiểm thử giao diện
- Kiểm thử bảo mật

Do giới hạn về thời gian và phạm vi, khóa luận chỉ trình bày thiết kế kiểm thử chức năng cho một số ca kiểm thử nổi bật.

Kiểm thử chức năng: trong thiết kế ca kiểm thử chức năng thường được chia làm hai loại thiết kế là: Positive test case và Negative test case.

Positive test case: là các trường hợp kiểm thử xem chức năng có hoạt động như mong muốn với các dữ liệu đúng.

Negative test case: là các trường hợp kiểm thử các chức năng với các dữ liệu đầu vào sai.

Áp dụng vào thiết kế chức năng đăng nhập cho hệ thống quản lí trẻ em ở trường mầm non trên, ta có hình 2-4 sau:

Positive test case	Negative test case
1. Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu đúng.	1 Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu không hợp lệ.
2. Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu đúng khi ấn phím Enter thay cho click vào button đăng nhập.	2. Kiểm tra chức năng đăng nhập với tài khoản sai và mật khẩu đúng.
3.Kiểm tra tài khoản đã đúng định của một email?	3. Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu không đúng.
4. Kiểm tra link quên mật khẩu có hoạt động đúng?	4. Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu để trống hoặc toàn ký tự trắng.
	5. Kiểm tra chức năng đăng nhập với tài khoản để trắng và nhập mật khẩu.
	6. Kiểm tra chức năng đăng nhập với tài khoản và mật khẩu cùng để trắng.

Hình 2-4. Thiết kế ca kiểm thử

Để thiết kế ca kiểm thử có thể bao quát đầy đủ các chức năng của chương trình, chúng ta áp dụng các kỹ thuật kiểm thử để xác định các dữ liệu đầu vào như sau:

Dữ liệu đúng: tài khoản và mật khẩu đúng, đã được đăng ký.

Dữ liệu sai:

- Tài khoản sai (không tồn tại, không đúng định dạng email,v.v.)
- Mật khẩu sai (mật khẩu không trùng với tài khoản,v.v.)
- Cả tài khoản và mật khẩu đều không đúng.

Dữ liệu để trống:

- Để trắng tài khoản
- Để trắng mật khẩu
- Để trắng cả mật khẩu và tài khoản.

Vì thời gian hạn chế, khóa luận chỉ trình bày một số ca kiểm thử thường gặp với chức năng đăng nhập hệ thống trong Hình 2-5 sau:

Test Case ID	**Test Case Description**	**Pre-Requisites**	**Steps**	**Execution Steps**	**Expected Results**	**Actual Results**	**Result**	**Note**
01	Check login successfully with valid username and passoword.	1.Login page displayed. 2. User must be registered already.	1 2 3	Enter correct username. Enter correct password. Click [Login] button.	Login successfully. Home page in displayed.	Login successfully. Home page in displayed.	Pass	
02	Check login failed with invalid username and valid password.	1.Login page displayed.	1 2 3	Enter incorrect username. Enter correct password. Click [Login] button.	Login failed. Show error message “Please enter valid username”.	Login failed. Show error message “Please enter valid username”.	Pass	

03	Check login failed with invalid password and valid username.	1.Login page displayed.	1 2 3	Enter correct username. Enter incorrect password. Click [Login] button.	Login failed. Show error message "Please enter valid password".	Login failed. Show error message "Please enter valid password".	Pass	
04	Check login failed with invalid username & password.	1.Login page displayed.	1 2	Enter incorrect username & password. Click [Login] button.	Login failed. Show error message "Please enter valid username and password".	Login failed. Show error message "Please enter valid username and password".	Pass	
05	Check login failed with emty username & password.	1.Login page displayed.	1 2	Do not enter username and password. Click [Login] button.	Error message is displayed to require user enter username and password.	Error message is displayed to require user enter username and password.	Pass	

06	Check that password must be masked	1.Login page displayed.	1 2	Enter username. Enter correct password	Password is shown as asterisk.	Password is shown as asterisk.	Pass	
07	Check lost your password link	1.Login page displayed.	1	Click on [Lost your password].	Redirect to password recovery page.	Do not redirect to password recovery page.	Fail	
08	Check login function when pressing Enter	1.Login page displayed.	1 2	Enter correct username and password. Press Enter button.	Login successfully. Home page is displayed.	Login successfully. Home page is displayed.	Pass	

Hình 2-5. Một số ca kiểm thử mẫu

BUG REPORTS

Project: XXX

Reported by: Tien,Nguyen Manh

Bug Name:	Page recovery password
Bug ID:	LG01
Date:	22-Mar-18
Assigned to:	**Developer-TEAM01**
Status:	New
Summary/Description:	Do not displayed when click on [Lost your password] link.
Environments (OS/Browser):	**Windows 10 /64bit/Core I5/ Ram 4GB/…**
Step to reproduce:	1.Click on [Lost your password].
Actual results:	Do not displayed recovery password page after click on [Lost your password] link.
Expected results:	Redirect to password recovery page when click on [Lost your password] link.
Severity:	Critical (S1)
Priority:	High (P1)
Attachment:	

Hình 2-6. Mẫu minh họa Bug Report

Trong đó:

Project: Tên dự án thực hiện.

Reported by: Tên người thực báo cáo.

Bug Name: Tên lỗi tìm được.

Bug ID: Mã lỗi.

Date: Ngày lập báo cáo.

Assigned to: Gán lỗi cho nhóm/ người sửa lỗi.

Status: Trạng của lỗi khi phát hiện.

Summary/Description: Miểu tả lỗi.

Environments (OS/Browser): Môi trường thực hiện kiểm thử.

Step to reproduce: Các bước thực hiện khi phát hiện ra lỗi.

Actual results: Kết quả thật khi thực hiện kiểm thử.

Expected results: Kết quả mong đợi sau khi sửa lỗi.

Severity: Cấp độ của lỗi.

Priority: Độ ưu tiên sửa lỗi.

Attachment: file đính kèm, miểu tả lỗi gặp phải(tệp, URL, ảnh,v.v.).

Để có thể sinh ca kiểm thử từ biểu đồ luồng dữ liệu, cần phải trải qua những bước cơ bản như sau:

Bước 1: Nhập biểu đồ luồng dữ liệu

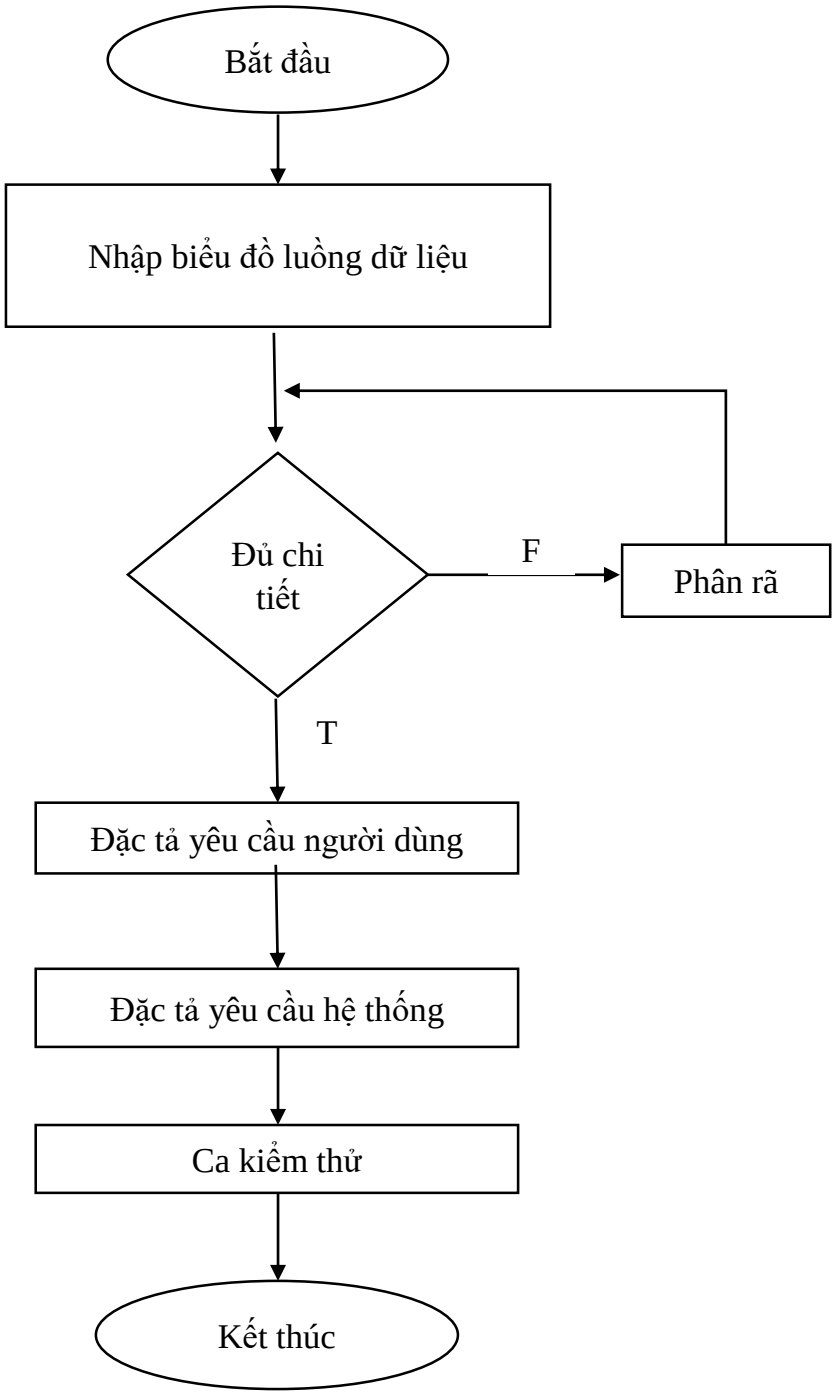
Bước 2: Phân rã tới mức đủ chi tiết

Bước 3: Viết đặc tả yêu cầu người dùng

Bước 4: Viết đặc tả yêu cầu hệ thống

Bước 5: Thiết kế ca kiểm thử

Có thể biểu diễn bằng sơ đồ khối như (Hình 2-7) sau:



Hình 2-7. Quy trình xây dự ca kiểm thử từ biểu đồ luồng dữ liệu

CHƯƠNG 3: ỨNG DỤNG KIỂM THỬ VỚI CÔNG CỤ RANOREX STUDIO

Trong phạm vi tìm hiểu khóa luận chỉ trình bày một trong những công cụ phổ biến được nhiều người dùng để tìm hiểu và thực nghiệm. Đó là công cụ Ranorex Studio. Đây là công cụ rất mạnh và tiện lợi có thể dùng để kiểm thử tự động tất cả các nền tảng bảo gồm web, deskop và mobile.

3.1 Giới thiệu Ranorex Studio

Ranorex Studio là một công cụ kiểm thử trả phí rất mạnh giúp đẩy nhanh công việc kiểm thử cho các ứng dụng dekop, web và di động.

Công cụ cung cấp cho kiểm thử viên bộ công cụ hoàn chỉnh để kiểm thử đầu cuối các ứng dụng. Ranorex cũng giúp chúng ta kiểm thử tự động các ứng dụng trên Windows, trên các thiết bị di động thật IOS/Android hoặc trên mô phỏng giả lập. Trên nên Web, Ranorex cho phép kiểm thử chéo trình duyệt cho Chrome, Firefox, Safari, Microsoft Edge và nhiều hơn nữa. Với Ranorex Studio kiểm thử viên sẽ tốn ít thời gian hơn cho việc giải quyết các vấn đề kiểm tra và đánh giá chất lượng của các ứng dụng.

3.2 Các thành phần của Ranorex Studio

Ranorex Studio chia làm 3 thành phần chính, mỗi phiên bản đều có vai trò cụ thể trong việc kiểm thử. Các thành phần đó là:

Deskop Testing: là công cụ con thuộc Ranorex Studio phát triển cho việc kiểm thử với các ứng dụng, hệ thống trên Windows mà không bị giới hạn cho dù nó dựa trên CEF, WPF, Java, .NET hay SAP. Ranorex Studio sẽ ngay lập tức nhận diện tất cả đối tượng trong ứng dụng đưa vào.

Để đảm bảo tận dụng tối đa thời gian của kiểm thử viên, Ranorex Studio cũng cho phép kiểm thử đồng thời các ứng dụng trên deskop trong nhiều môi trường - sử dụng các cấu hình của hệ thống khác nhau bằng công cụ Ranorex

Remote. Kiểm thử viên cũng có thể tiếp tục làm việc trên máy tính của mình trong thời gian chờ đợi nhận báo cáo kết quả kiểm thử. Bên cạnh đó, Ranorex Remote cũng rất mạnh trong làm việc nhóm. Kiểm thử viên trong nhóm có thể truy cập và triển khai kiểm thử cho cùng một ứng dụng, cũng như nhận được tất cả các báo cáo kiểm thử từ xa để theo dõi tình trạng và hiệu suất của dự án.

Web Testing: là công cụ cho phép kiểm thử viên tạo kiểm thử một lần và chạy nó trên nhiều trình duyệt và phiên bản trình duyệt.

Để đảm bảo trải nghiệm người dùng cuối là hoàn hảo cũng như tiết kiệm thời gian, chi phí, kiểm thử viên chỉ cần ghi lại kịch bản kiểm thử một lần và sau đó sử dụng lại trong Firefox, IE, Edge, Chrome và Safari mà không cần sửa đổi.

Mobile Testing: cho phép kiểm thử trên tất cả các thiết bị mobile thật hoặc chế độ giả lập để thuận tiện hơn trong việc sửa các lỗi gặp phải. Với cơ chế nhận dạng đối tượng, Ranorex giúp bạn dễ dàng xác nhận và xác minh vào quy trình kiểm thử các ứng dụng di động của mình. Phương pháp này cho phép bạn kiểm tra số lượng lớn các thuộc tính điều khiển.

Với thiết bị giả lập phong phú có sẵn từ máy tính bảng, điện thoại thông minh và thiết bị đeo. Ranorex đảm bảo các ứng dụng di động hoạt động trên tất cả các thiết bị với thực hiện kiểm thử di động song song trên các thiết bị IOS và Android khác nhau.

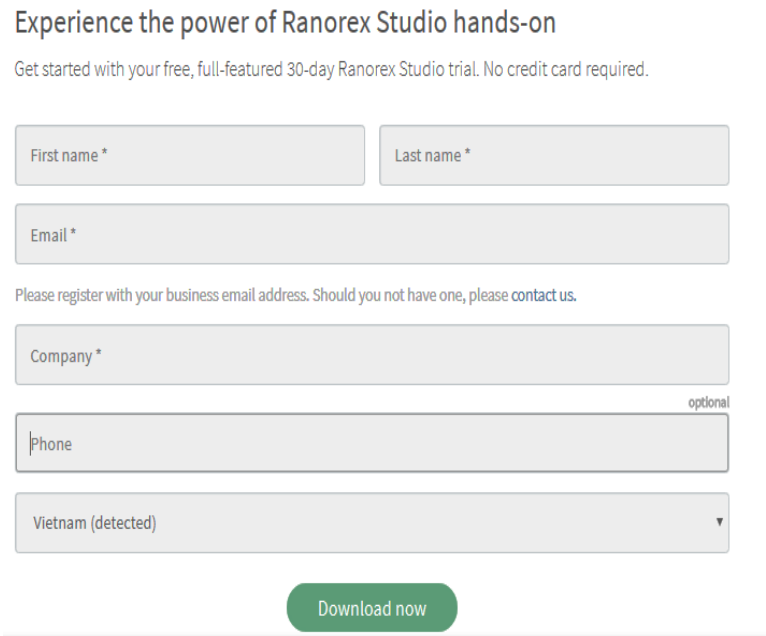
3.3 Cài đặt Ranorex Studio

Vì đây là công cụ trả phí, nên trong khóa luận này em sẽ trình bày cách cài đặt với phiên bản dùng thử 30 ngày của Ranorex Studio dành cho kiểm thử ứng dụng Desktop.

Bước 1: truy cập vào website:

https://www.ranorex.com/windows-desktop-test-automation/

Bước 2: Click vào Download free trial, sau đó điền thông tin đăng kí để nhận được link tải phần mềm, Ranorex Studio yêu cầu phải đăng kí bằng email công ty cơ quan mới cho phép dùng thử, đối với các email cá nhân sẽ không được hỗ trợ.

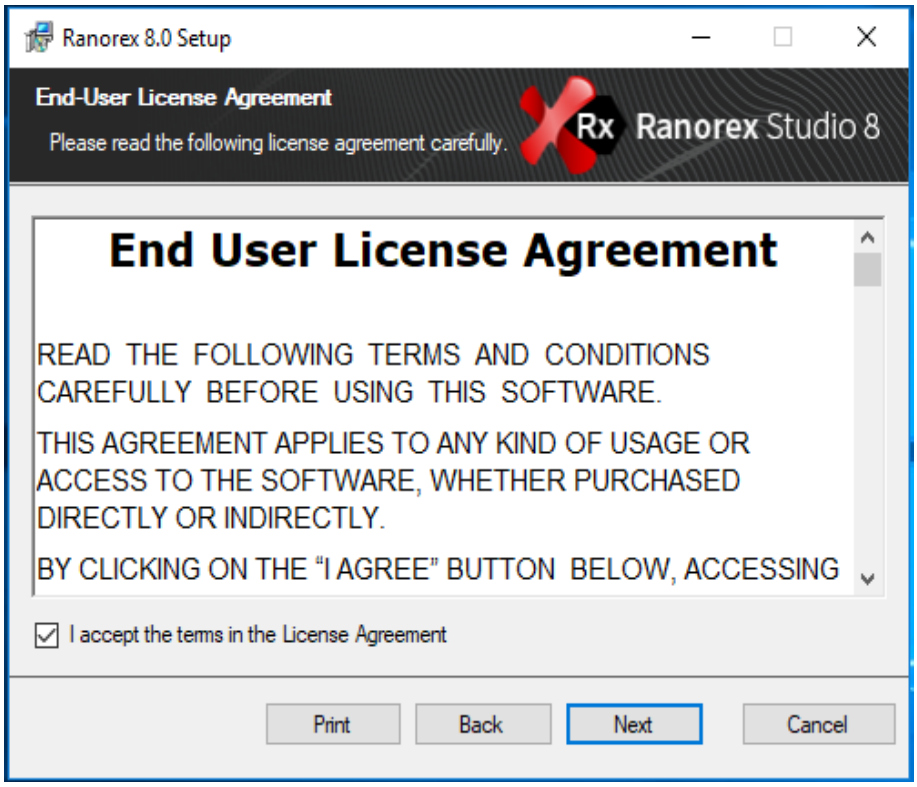


Hình 3-1. Cài đặt Ranorex Studio

Bước 3: Sau khi tải về xong sẽ xuất hiện chương trình cài đặt như Hình 3-2:

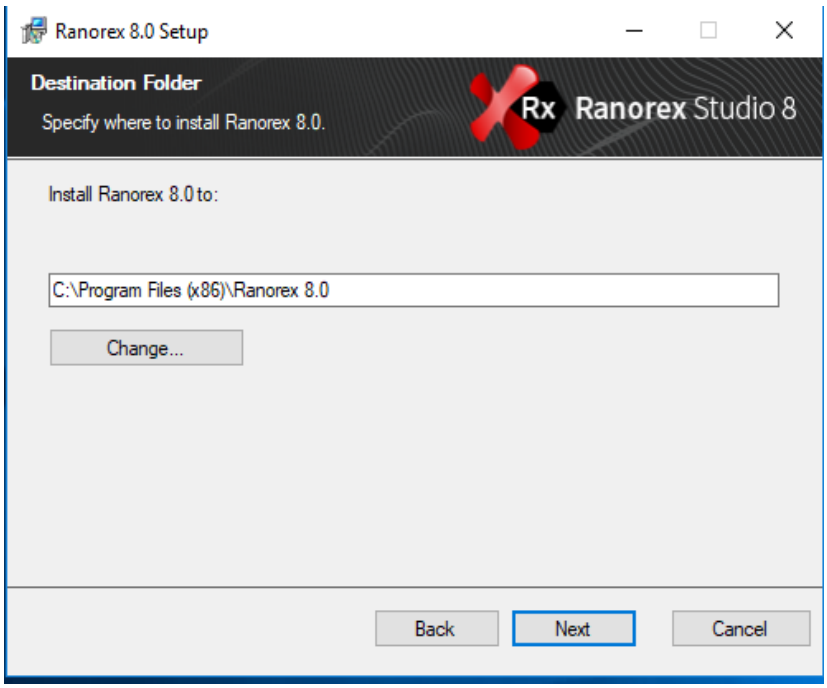
Hình 3-2. Cài đặt Ranorex Studio

Click vào Next để tiếp tục cài đặt:



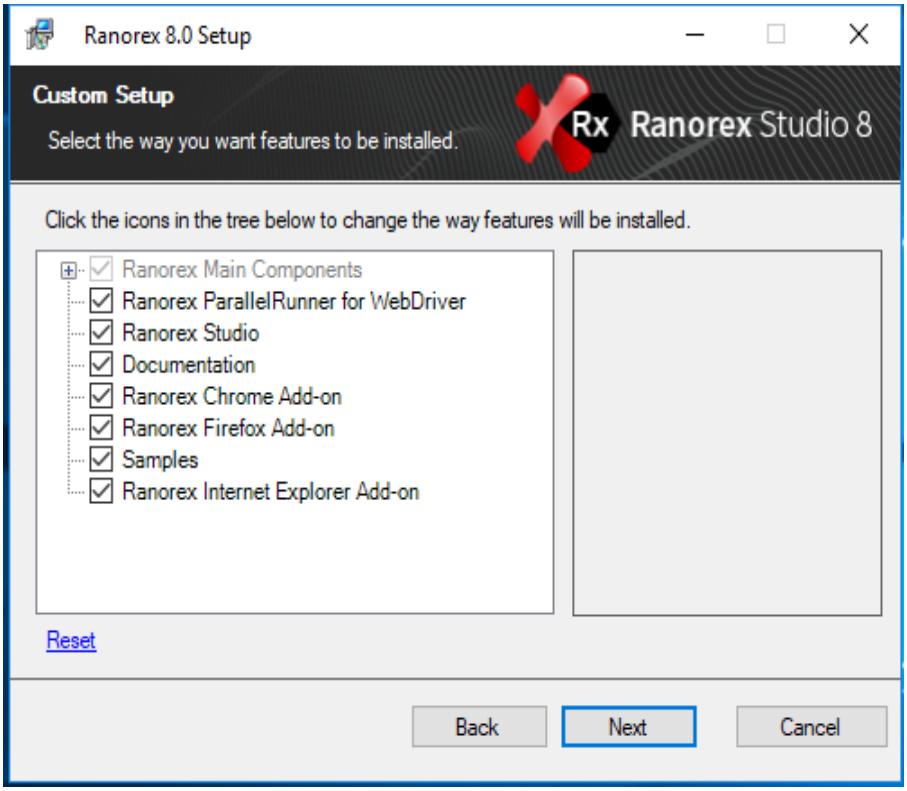
Hình 3-3. Cài đặt Ranorex Studio

Click vào đồng ý các điều khoản và chính sách sau đó click Next:



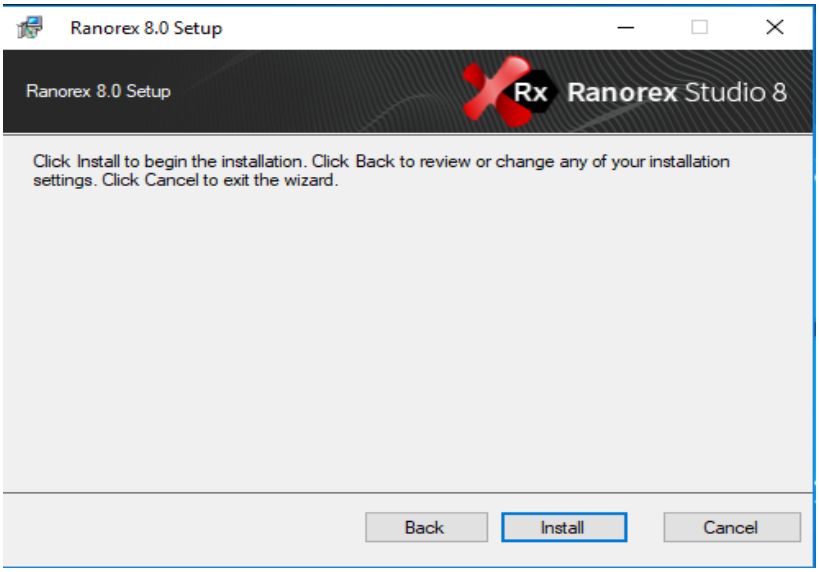
Hình 3-4. Cài đặt Ranorex Studio

Ranorex mặc định tại địa chỉ C:\Program Files(x86)\... bạn có thể thay đổi tùy đích cá nhân sử dụng. Sau đó click Next để tiếp tục:



Hình 3-5. Cài đặt Ranorex Studio

Tại đây là lựa chọn các công cụ con thêm của Ranorex phát triển, để mặc định chọn hết hoặc bỏ tích các công cụ còn lại chỉ để Ranorex Studio.



Hình 3-6. Cài đặt Ranorex Studio

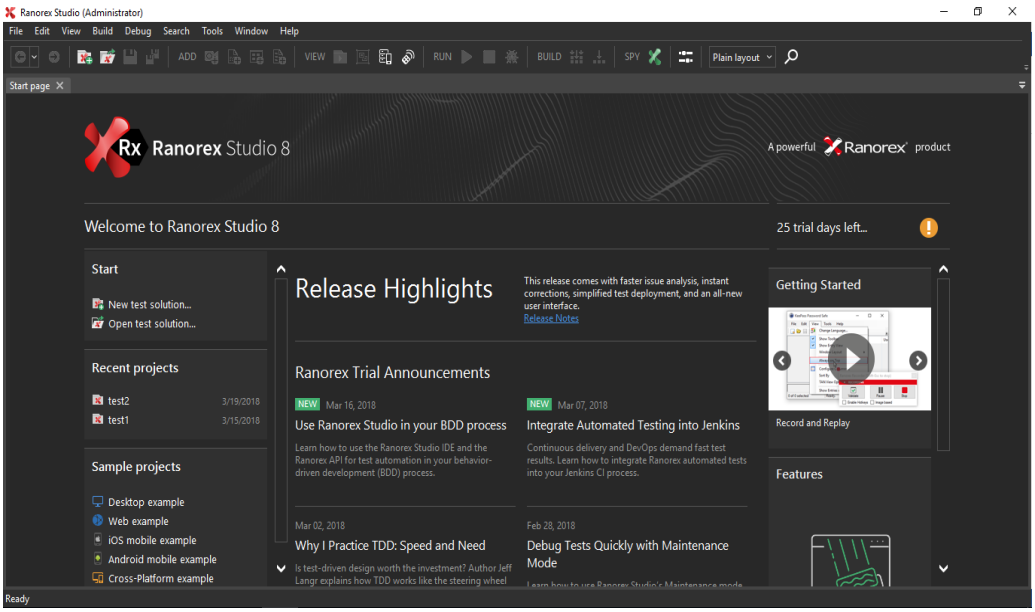
Click Install để cài đặt chương trình:

Hình 3-7. Cài đặt Ranorex Studio

Click Finish để hoàn thành chương trình cài đặt Ranorex Studio.

Hình 3-8. Cài đặt Ranorex Studio

Chọn “Start free trial ” để sử dụng 30 ngày chương trình miễn phí.



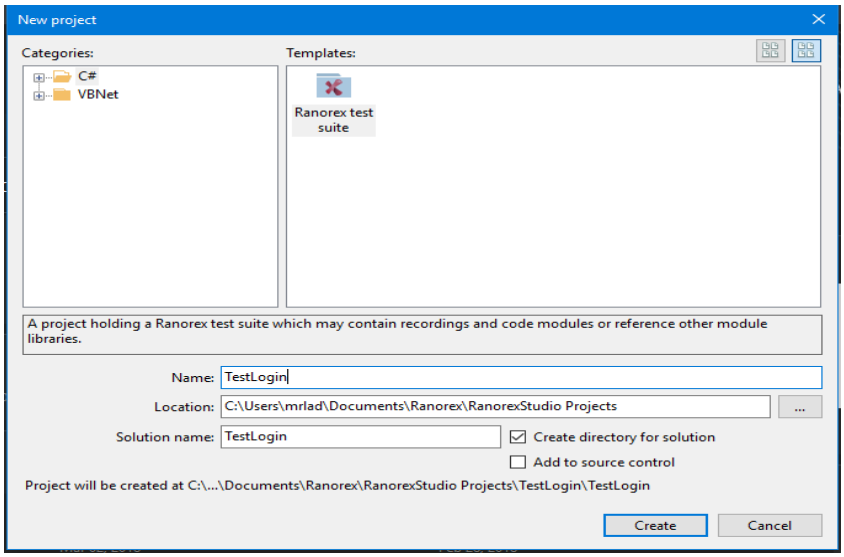
Hình 3-9. Màn hình làm việc Ranorex Studio

Màn hình làm việc chính của Ranorex Studio.

3.3.1 Ứng dụng

Khóa luận thực hiện kiểm thử với chương trình Calculator trên windows.

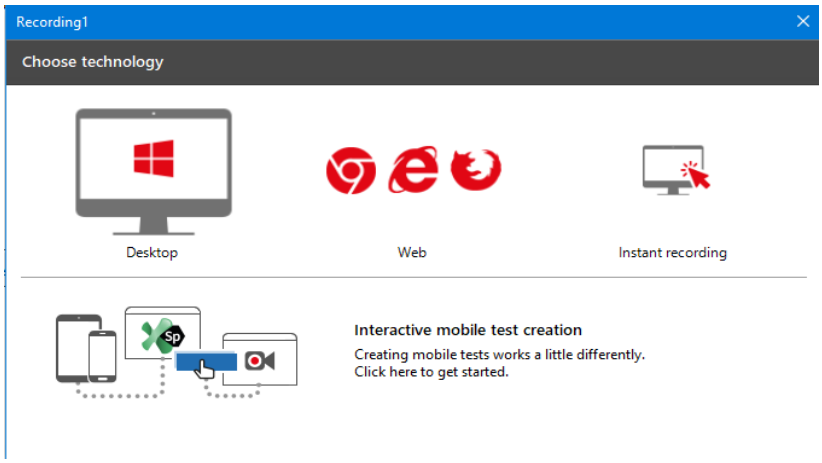
B1: Tạo project Test.



Hình 3-10. Thực hành trên công cụ Ranorex Studio

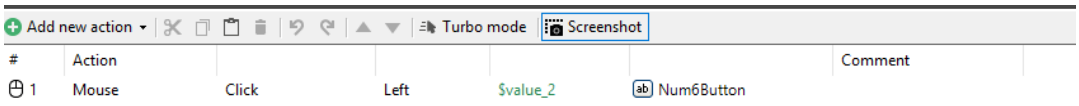
Bước 2: Tạo Record để chuẩn bị kịch bản kiểm thử.

Chon Instant Recording để ghi lại hoạt động làm cơ sở thiết kế ca kiểm thử.



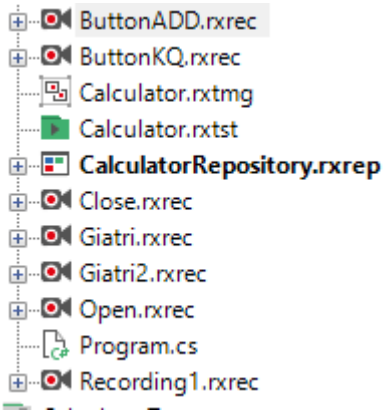
Hình 3-11. Thực hành trên công cụ Ranorex Studio

Thay đổi giá trị biến để kiểm thử với các dữ liệu khác:



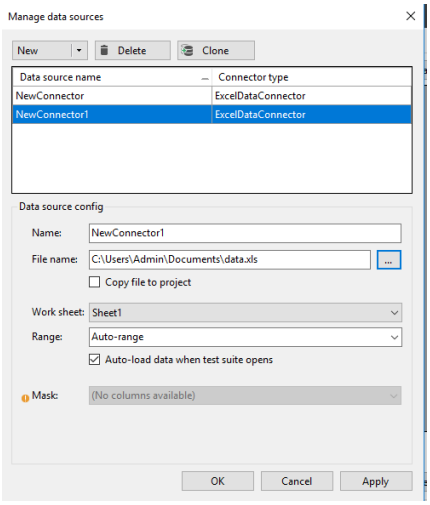
Hình. 3-12. Thay đổi biến giá trị

Tại đây có thể nhóm các hành động để tái sử dụng cũng như thay đổi các trường dữ liệu liệu cho kiểm thử tự động về sau.



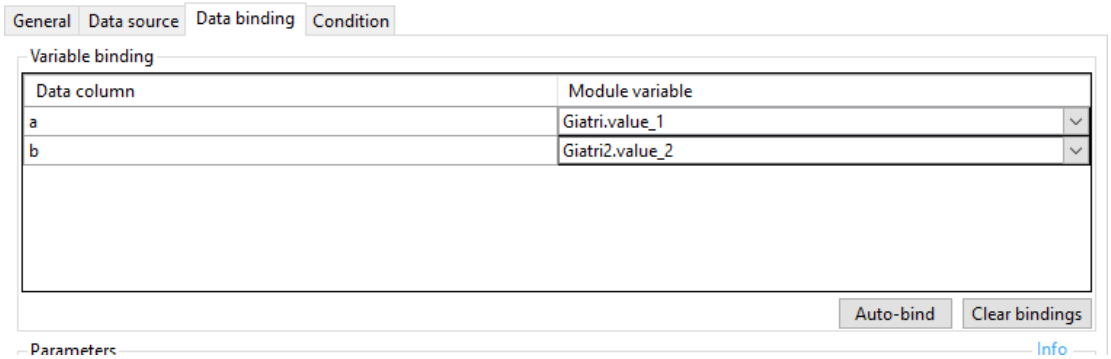
Hình. 3-13.Nhóm các hoạt động ghi

Kết nối cơ sở dữ liệu (file excel, sql,v.v.) thay đổi các giá trị biến kiểmthử.



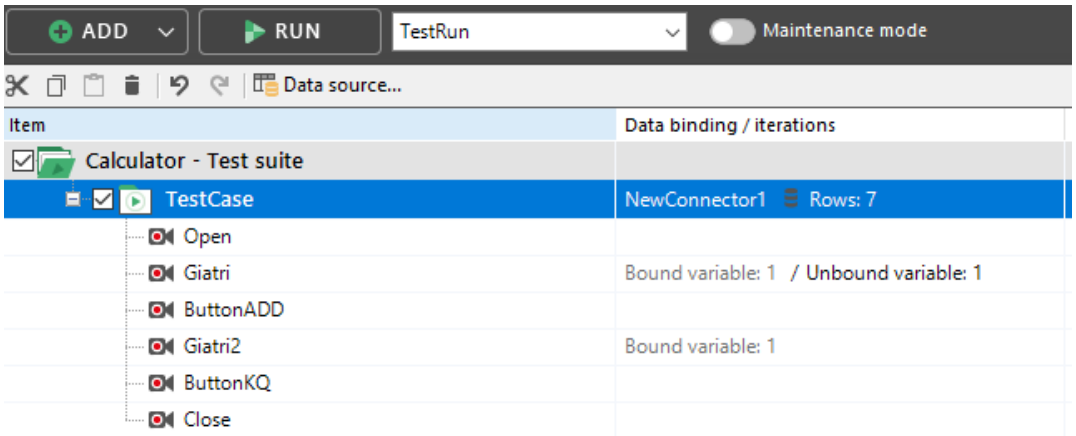
Hình. 3-14. Kết nối dữ liệu kiểm thử

Gán giá trị cần thay đổi với dữ liệu đầu vào.



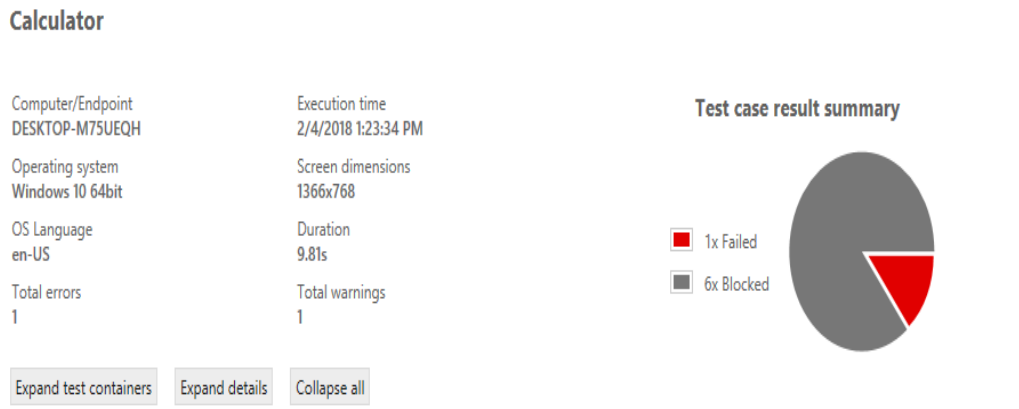
Hình. 3-2.Gán giá trị thay đổi

Thực hiện chạy để kiểm tra với các giá trị khác trong dữ liệu đầu vào:



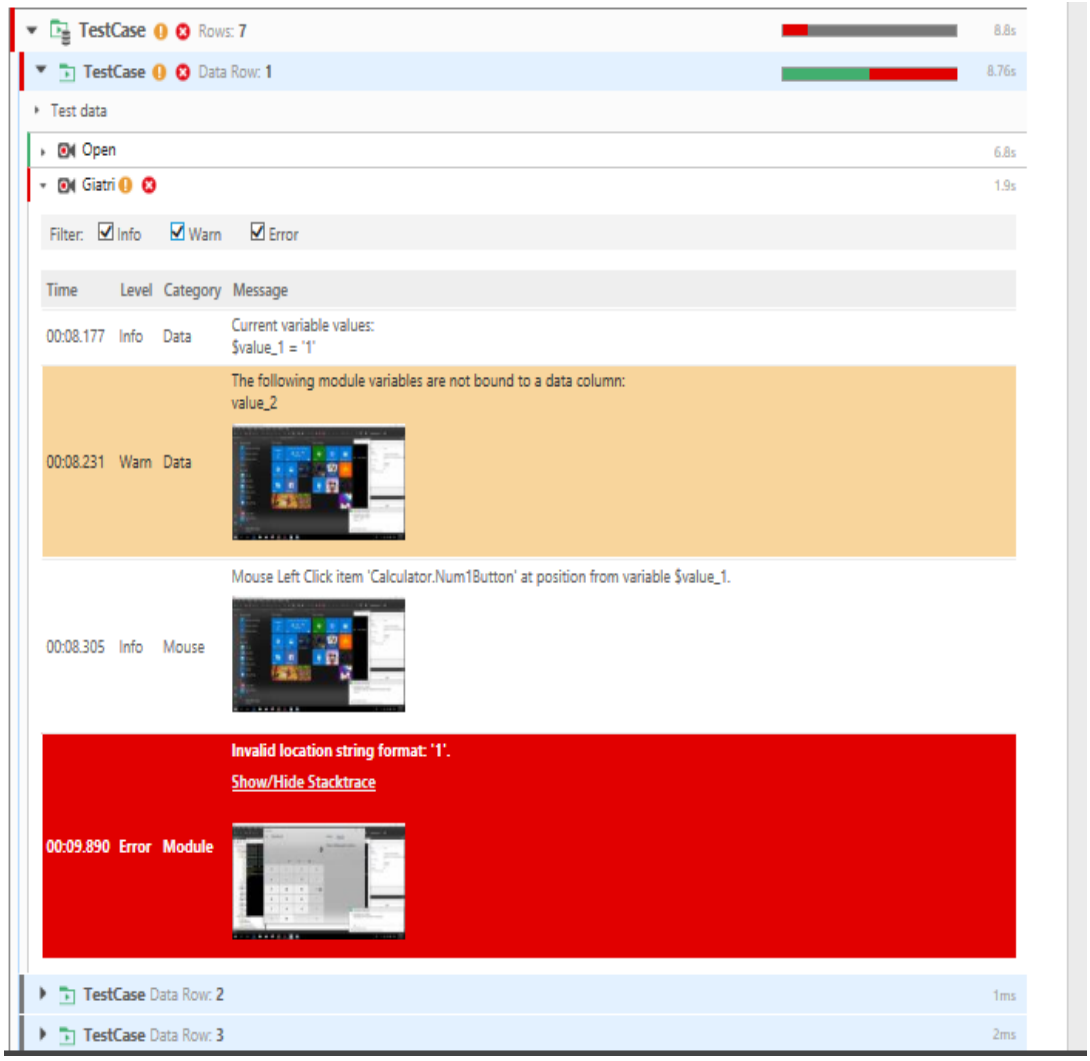
Hình. 3-15. Thực hiện ca kiểm thử

Kết quả sau khi chạy ca kiểm thử:



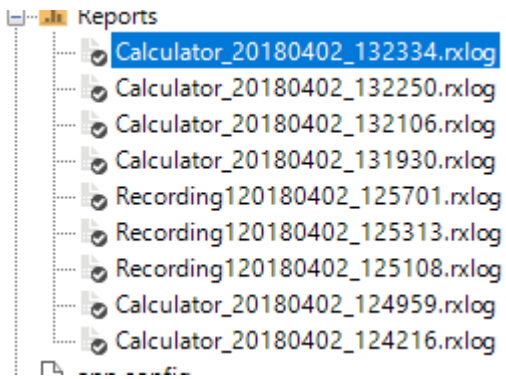
Hình. 3-16.Báo cáo kết quả ca kiểm thử

Báo cáo chi tiết:



Hình. 3-17. Báo cáo kết quả ca kiểm thử chi tiết

Các bản báo cáo của từng ca kiểm thử khác:



Hình. 3-18. Tập báo cáo từng ca kiểm thử

Mã nguồn khởi động chương trình cho việc kiểm thử được tự động sinh ra sau các hoạt động.

```
 * Created by Ranorex
 * User: Admin
 * Date: 3/27/2018
 * Time: 3:22 PM
  * To change this template use Tools > Options > Coding > Edit standard headers.
using System;
using System.Threading;
using System.Drawing;
using System.Collections.Generic;
using System.Text.RegularExpressions;
using WinForms = System.Windows.Forms;
using Ranorex;
using Ranorex.Core;
using Ranorex.Core.Reporting;
using Ranorex.Core.Testing;
```

Mã nguồn chính của ca kiểm thử:

```
namespace Calculator
{
   class Program
   {
      [STAThread]
      public static int Main(string[] args)
      {   // Uncomment the following 2 lines if you want to automate Windows apps
         // by starting the test executable directly
         //if (Util.IsRestartRequiredForWinAppAccess)
         //    return Util.RestartWithUiAccess();
         Keyboard.AbortKey = System.Windows.Forms.Keys.Pause;
         int error = 0;
         try
         {
            error = TestSuiteRunner.Run(typeof(Program), Environment.CommandLine);
         }
         catch (Exception e)
         {
            Report.Error("Unexpected exception occurred: " + e.ToString());
            error = -1;
         }
         return error;
      }
   }
}
```

KẾT LUẬN

Kiểm thử phần mềm luôn là vấn đề hết sức quan trọng đối với các tổ chức phát triển phần mềm. Trong đó, kĩ thuật xây dựng ca kiểm thử từ biểu đồ luồng dữ liệu không phải là mới nhưng luôn là một phần quan trọng của kiểm thử phần mềm trong các dự án. Trong khuôn khổ của khóa luận do thời gian và kinh nghiệm còn hạn chế nên có những phần của khóa luận chưa được đào sâu nghiên cứu.

Sau một thời gian thực hiện khóa luận dưới sự hướng dẫn của thạc sĩ Nguyễn Trịnh Đông, khóa luận đã thực hiện tốt được các mục tiêu đề ra và đạt được những kết quả sau:

Kết quả đạt được:

- Trình bày cơ bản và chính xác các vấn đề về phần mềm, công nghệ phần mềm, lỗi phần mềm và các vấn đề liên quan đến kiểm thử phần mềm.
- Tìm hiểu và nắm được phương pháp xây dựng kĩ thuật ca kiểm thử từ biểu đồ luồng dữ liệu.
- Giới thiệu công cụ Ranorex Studio, các thao tác cơ bản sử dụng.
- Áp dụng các kiến thức tìm hiểu thực hiện kiểm thử chức năng với công cụ Ranorex Studio.
- Khóa luận là một tài liệu xúc tích tổng hợp được các vấn đề chính của phần mềm và kiểm thử phần mềm.
- Nâng cao khả năng đọc hiểu tài liệu Tiếng Anh.
- Bổ sung và rèn luyện thêm kỹ năng sử dụng phần mềm Word và Powerpoint.

Hạn chế:

Mặc dù đã cố gắng hết sức trong thời gian thực hiện khóa luận nhưng với kinh nghiệm còn hạn chế nên khóa luận không tránh khỏi những thiếu sót:

- Chỉ trình bày được những khái niệm cơ bản nhất, chưa đi sâu vào lý thuyết phần mềm và kiểm thử phần mềm.
- Chỉ tìm hiểu được một trong ba công cụ của bộ Ranorex Studio. Còn hai bộ công cụ là Web Testing và Mobile Testing chỉ giới thiệu sơ qua.
- Sự áp dụng những kiến thức tìm hiểu được mới chỉ dừng lại ở một chương trình nhỏ, mà vẫn chưa thử áp dụng cho các bài toán hay ứng dụng lớn.

TÀI LIỆU THAM KHẢO

[1]. Giáo trình kiểm thử phần mềm – Phạm Ngọc Hùng, Trương Anh Hoàng và Đặng Văn Hưng (Tháng 1 năm 2014) .

[2]. Kỹ nghệ phần mềm – Nguyễn Văn Vị và Nguyễn Việt Hà.

[3]. Bách khoa toàn thư mở Wikipedia.

[4]. IEEE Standard Glossary of Software Engineering Terminology.

[5]. The Art of Software Testing.

[6]. Websites: www.testing.vn, www.testingvn.com, https://viblo.asia/.

[7]. Website: www.ranorex.com.