BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC ĐÀ NẴNG

THÂN THỊ TÂM

NGHIÊN CỨU ỨNG DỤNG RELEVANTCODES ĐỂ XÂY DỰNG FRAMEWORK TRONG KIỂM THỬ TỰ ĐỘNG

Chuyên ngành: KHOA HỌC MÁY TÍNH

Mã số: 60.48.01

TÓM TẮT LUẬN VĂN THẠC SĨ KỸ THUẬT

ĐÀ NẴNG - Năm 2012

Công trình được hoành thành tại

ĐẠI HỌC ĐÀ NẴNG

Người hướng dẫn khoa học: TS. NGUYỄN THANH BÌNH

Phản biện 1: TS. HUỲNH CÔNG PHÁP

Phản biện 2: PGS.TS. ĐOÀN VĂN BAN

Luận văn được bảo vệ trước Hội đồng chấm Luận văn tốt nghiệp thạc sĩ kỹ thuật tại Đại học Đà Nẵng vào ngày 3 tháng 3 năm 2012.

Có thể tìm hiểu luận văn tại:

- Trung tâm Thông tin - Học liệu, Đại học Đà Nẵng
- Trung tâm Học liệu, Đại học Đà Nẵng

MỞ ĐẦU

1. Lý do chọn đề tài

Kiểm thử tự động (KTTĐ) là một lĩnh vực nhằm thu hút được lợi ích tối đa với nỗ lực tối thiểu đối với các công việc lặp đi lặp lại. Lợi ích tối đa ở đây chính là khả năng gia tăng hiệu quả các nguồn nhân lực, gia tăng độ bao phủ của việc kiểm thử, nâng cao chất lượng và độ tin cậy của phần mềm.

KTTĐ đòi hỏi một phương pháp tiếp cận được xác định, dựa trên một framework toàn diện, để tận dụng và đạt được lợi ích tối đa trong giai đoạn kiểm thử của một quy trình sản xuất phần mềm.

Một framework KTTĐ có nhiệm vụ chính trong việc định nghĩa các khuôn mẫu diễn đạt được đúng yêu cầu mong đợi, tạo ra một cơ chế kiểm tra, kiểm thử phần mềm ứng dụng, thực thi việc kiểm thử và thông báo kết quả kiểm thử.

Để đáp ứng nhu cầu ngày càng phát triển của một quá trình KTTĐ, cần có một hệ thống với một khuôn mẫu thống nhất có tính mở rộng cao để bắt kịp với các nhu cầu thay đổi trong quy trình phát triển phần mềm. Framework được thiết kế để đáp ứng nhu cầu này, chỉ cần xây dựng thêm các mô-đun vào framework tương ứng với sự thay đổi của ứng dụng.

Trong phạm vi đề tài tài này, tôi muốn khai thác mã nguồn mở cho việc nghiên cứu và triển khai thử nhiệm. Cách tiếp cận ở đây là để thúc đẩy việc sử dụng lại mã nguồn có sẵn, tân dụng tối ưu các điểm mạnh của công cụ mã nguồn mở nhằm đêm lại năng suất cao hơn.

Đối với mã nguồn mở, các trường hợp kiểm thử tương ứng các chức năng của ứng dụng sẽ được quản lý rời rạc được mô tả thông qua việc sử dụng các từ khóa. Công cụ mã nguồn mở mà tôi sử dụng để xây dựng framework cho các ứng dụng web có tên là RelevantCodes.

Trên đây là lý do chọn đề tài 'Nghiên cứu ứng dụng RelevantCodes để xây dựng framework trong kiểm thử tự động'.

2. Mục tiêu và nhiệm vụ nghiên cứu

Mục tiêu của đề tài là nghiên cứu mã nguồn mở RelevantCodes (RC) và xây dựng framework ứng dụng cho KTTĐ.

Nhiệm vụ nghiên cứu của đề tài là trình bày sơ lược về quy trình kiểm thử tự động, đánh giá một số các framework có sẵn sử dụng cho quy trình kiểm thử tự động, cuối cùng là ứng dụng RC để xây dựng và cải tiến nó thành framework RC+ với cấu trúc mới.

3. Đối tượng nghiên cứu

Để thực hiện được mục tiêu và nhiệm vụ của đề tài, luận văn cần đề cập đến các đối tượng nghiên cứu sau đây:

- Quy trình kiểm thử tự động
- Các công cụ kiểm thử tự động
- Framework kiểm thử tự động
- Tìm hiểu framework RelevantCodes trong kiểm thử tự động

4. Phạm vi nghiên cứu

- Nghiên cứu phát triển mã nguồn mở RelevantCodes để cải tiến thành một cấu trúc framework mới có thể ứng dụng được trong thực tế
- Ứng dụng công cụ RelevantCodes đã cải tiến cho các các ứng dụng web thông thường (HTML)
- Phát triển thêm một số chức năng vào thư viện dùng chung

5. Phương pháp nghiên cứu

- Tìm hiểu về quy trình KTTĐ
- Tìm hiểu tổng quan về framework cho quy trình KTTĐ
- Phân tích, đánh giá các chức năng, lợi ích, kiến trúc của RelevantCodes
- Phát triển RelevantCodes để triển khai một framework cụ thể
- Kiểm tra, thử nghiệm và đánh giá kết quả

6. Những phương tiện công cụ để có thể triển khai

- Sử dụng framework mã nguồn RelevantCodes để phát triển ứng dụng
- Sử dụng công cụ Quick Test Pro - phiên bản thử nghiệm (trial) - làm phương tiện để triển khai phát triển RelevantCodes

7. Ý nghĩa khoa học và thực tiễn của đề tài

Phần nghiên cứu lý thuyết cũng như đánh giá các framework sẽ cung cấp một cách khái quát và phần nào giúp người đọc hiểu được lợi ích, tầm quan trọng của quy trình KTTĐ.

Kết quả nghiên cứu cũng sẽ hướng sự quan tâm của người đọc đến các lợi ích cũng như tầm quan trọng của mã nguồn mở trong việc xây dựng các framework cho kiểm thử phần mềm nói riêng và các ứng dụng, tiện ích, hiệu quả cũng như việc tiết kiệm chi phí khi dùng mã nguồn mở nói chung.

8. Đặt tên đề tài

Tên đề tài là 'Nghiên cứu ứng dụng RelevantCodes để xây dựng framework trong kiểm thử tự động'.

9. Bố cục luận văn

Luận văn được tổ chức thành ba chương.

Chương 1, tiêu đề là '*Kiểm thử tự động*', trình bày tổng quan về kiểm thử tự động, các công cụ kiểm thử tự động và framework kiểm thử tự động.

Chương 2, tiêu đề chương là '*Giải pháp kiểm thử tự động với RelevantCodes và QuickTestPro*'. Chương này trình bày cụ thể về công cụ kiểm thử QTP và framework mã nguồn mở RC. Bên cạnh đó, trình bày sự kết hợp của QTP và RC cùng mối liên hệ và sự phụ thuộc của hai công cụ này.

Chương cuối cùng là chương 3 có tiêu đề là '*Cải tiến RelevantCodes và ứng dụng vào kiểm thử tự động*', trình bày hướng giải quyết những vấn đề hạn chế của RC và triển khai cải tiến RC thành RC+, sau đó trình bày cấu trúc của framework RC+, thử nghiệm và nhận xét đánh giá RC+.

CHƯƠNG 1
KIỂM THỬ TỰ ĐỘNG

Chương đầu tiên của luận văn trình bày sơ lược các vấn đề xung quanh KTTĐ, bao gồm tổng quan về KTTĐ, tổng quan về công cụ KTTĐ, và framework KTTĐ.

1.1. Tổng quan về kiểm thử tự động

Phần này sẽ trình bày bao quát các vấn đề liên quan đến vấn đề kiể thử tự động một cách bao quát bao gồm: giới thiệu một số định nghĩa về kiểm thử tự động, mục tiêu, yêu cầu thiết yếu của kiểm thử tự động, phương pháp quản lý vòng đời của quy trình kiểm thử tự động (trong phạm vi đề tài, phương pháp được sử dụng là phương pháp luận vòng đời kiểm thử tự động ATLM - Automated Testing Lifecycle Methodology - ATLM), các cấp độ của kiểm thử tự động, phân loại kiểm thử tự động, Xác định các đối tượng có thể được kiểm thử tự động và cuối cùng đề cập đến những mong đợi sai lầm về kiểm thử tự động.

1.1.1. Giơi thiệu

Kiểm thử tự động là việc sử dụng các phần mềm để kiểm soát việc thực hiện các thử nghiệm, so sánh kết quả thực tế kết quả dự đoán, thiết lập các điều kiện tiên quyết cho các thử nghiệm, và các chức năng kiểm cũng như báo cáo kết quả thử nghiệm. Thông thường, KTTĐ bao hàm việc việc tự động hoá một quy trình làm việc bằng tay đã được sử dụng như một quá trình kiểm thử chính thức.

1.1.2. Mục tiêu, yêu cầu thiết yếu của kiểm thử tự động

Mục tiêu của KTTĐ là làm giảm các hoạt động kiểm thử thủ công và các hoạt động kiểm thử dư thừa bằng cách sử dụng một giải pháp có hệ thống để đạt được một phạm vi và độ bao phủ các trường hợp kiểm thử tốt hơn, giảm chi phí và thời gian kiểm thử trong suốt vòng đời phần mềm. Qua đó, nâng cao chất lượng, tăng độ tin cậy, tăng tốc độ làm việc và hiệu quả của quá trình kiểm thử, đạt được các tiêu chí kiểm thử, giải phóng cho các kỹ sư kiểm thử phần mềm thoát khỏi việc thực hiện kiểm thử cách tẻ nhạt và lặp lại nhàm chán.

Để đạt được những mục tiêu đó, quy trình KTTĐ đặt ra nhiều yêu cầu thiết yếu. Việc phải thành lập một đội ngũ kỹ sư chuyên dụng cho KTTĐ với một kế hoạch và chiến lược rõ ràng, các kỹ sư phải giỏi kĩ năng lập trình. Các công cụ kiểm thử phải phù hợp với chiến lược đặt ra, phù hợp với ngân sách dành riêng cho KTTĐ và tiến độ của dự án. Việc bảo trì các ca kiểm thử cũng là một yêu cầu thiết yếu của KTTĐ. Việc kiểm thử phải được thực thi nhiều lần, do đó yêu cầu phải có một quy trình phát triển cụ thể, song song đó, nên có quy trình kiểm thử tại chỗ – chủ yếu là thủ công. Cuối cùng, cần phải xem xét đến các chi phí liên quan đên chi phí của công cụ kiểm thử và chi phí đào tạo nếu có.

1.1.3. Quản lý vòng đời của quy trình KTTĐ

Ngày nay, các chuyên gia phần mềm đang phải đối mặt với những thách thức về việc xây dựng các hệ thống với ít nhân lực, tài nguyên trong một khoảng thời gian ngày càng bị thu hẹp. Các công ty không chỉ muốn kiểm thử phần mềm một cách đầy đủ, mà còn đòi

hỏi nhanh chóng và triệt để nhất. Để thực hiện được mục tiêu này, các tổ chức đang chuyển sang KTTĐ. Khi quyết định điều đó, có thể họ còn chưa biết đến một công cụ KTTĐ nào. Phương pháp luận vòng đời KTTĐ (Automated Testing Lifecycle Methodology - ATLM) sẽ cung cấp hướng dẫn về quy trình KTTĐ.

ATLM là một phương pháp hướng cấu trúc để đảm bảo thực hiện thành công KTTĐ. Phương pháp này có cấu trúc liên quan đến một quá trình gồm nhiều giai đoạn hỗ trợ các hoạt động chi tiết và liên quan đến nhau, nó hỗ trợ phát triển các thiết kế kiểm thử, phát triển và thực thi các ca kiểm thử, quản lý các dữ liệu kiểm thử. Nó cũng hỗ trợ việc quản lý tài liệu, theo dõi, cho phép nhóm kiểm thử báo cáo các lỗi/sự cố của hệ thống.

1.1.4. Các cấp độ của kiểm thử tự động

Các cấp độ của KTTĐ có quy trình, giải pháp, khả năng hỗ trợ và được phân loại theo mục đích kiểm thử khác nhau. Có bốn cấp độ chính là quản lý KTTĐ, thực thi KTTĐ, tạo ra các ca KTTĐ và cuối cùng là tối ưu KTTĐ.

1.1.5. Phân loại kiểm thử tự động

Có nhiều loại hình KTTĐ khác nhau, luận văn đề cập đến bốn loại KTTĐ được phân loại dựa trên các kĩ thuật KTTĐ tương ứng cho từng loại, bao gồm kiểm thử giao diện đồ họa, kiểm thử hướng mã nguồn, tự động phát sinh các ca kiểm thử, tự động phát sinh bộ dữ liệu kiểm thử.

1.1.6. Xác định các đối tượng có thể được kiểm thử tự động

Việc lựa chọn các chức năng để thực hiện kiểm thử tự động có ảnh hưởng quyết định đến sự thành công của chiến lược kiểm thử tự động. Nên tránh chọn các chức năng không ổn định hoặc đang trong quá trình thay đổi.

1.1.7. Những mong đợi sai lầm về kiểm thử tự động

Công cụ KTTĐ có thể hỗ trợ tất cả các yêu cầu kiểm thử phần mềm: không phải vậy, không có công cụ nào hỗ trợ tất cả các công việc của tester được cả. Việc sử dụng các công cụ KTTĐ không dễ dàng như tưởng tượng. Có thể thiết lập kế hoạch kiểm thử để thực thi việc kiểm thử phần mềm mà không cần sự can thiệp thủ công nào. Mọi thứ đều có thể kiểm thử tự động được. Các nỗ lực kiểm thử, cũng như tiến độ kiểm thử sẽ được rút ngắn và giảm ngay lập tức, hay KTTĐ sẽ tiết kiệm, làm giảm nguồn lực và bù trừa cho các yêu cầu hay thiết kế tồi.

1.2. Tổng quan về công cụ kiểm thử tự động

Các công cụ KTTĐ được thiết kế đặc biệt gắn với một số môi trường kiểm thử cụ thể. Chẳng hạn như: công cụ cho ứng dụng Windows, công cụ cho ứng dụng web, v…v… Nó đóng vai trò điều khiển quy trình kiểm thử tự động, có thể trợ giúp tự động hoá các tác vụ như cài đặt sản phẩm, tạo ra bộ dữ liệu kiểm thử, tương tác giao diện đồ họa, phát hiện các vấn đề về đăng nhập, v…v…, mà không nhất thiết phải tự động trông mô hình end-to-end.

Nhiều công cụ kiểm thử tự động đã có sẵn trên thị trường. Một số dành cho ứng dụng web, một số dụng cho ứng dụng window, một số để kiểm thử cơ sở dữ liệu…

Trong phạm vi đề tài, tôi đề cập đến một số loại công cụ KTTĐ dựa trên các khía cạnh liên quan đến quy trình phần mềm, được chia thành 10 loại công cụ chính: công cụ quản lý yêu cầu, công cụ quản lý cấu hình, công cụ lập kế hoạch và quản lý các ca kiểm thử, công cụ phát sinh bộ dữ liệu kiểm thử, công cụ kiểm thử giao diện người dùng, công cụ kiểm thử hiệu năng, công cụ đo lường độ bao phủ của mã nguồn, công cụ phân tích dựa trên các quy luật, công cụ theo dõi các lỗi của phần mềm, và cuối cùng là công cụ gỡ rối.

1.3. Tổng quan về framework kiểm thử tự động

Với kiểm thử tự động, chúng ta quan tâm đến những sự thay đổi của hệ thống cần được kiểm thử qua mỗi phiên bản. Nhưng chúng không thể theo kịp trong trường hợp có đa dạng cầu hay sự thay đổi yêu cầu. Trong trường hợp này, cần một khuôn khổ thống nhất có thể kết hợp với công cụ KTTĐ để đạt được thành công cao hơn, ta gọi đó là framework KTTĐ.

Phần nội dung này sẽ trình bày một cách tổng quan về framework KTTĐ bao gồm các khái niệm về framework, phân loại framework, các chức năng, mục tiêu cũng như nhiệm vụ của framework.

1.3.1. Định nghĩa framework

Một framework kiểm thử tự động là một tập hợp các giả định, các khái niệm và công cụ được cung cấp để hỗ trợ cho quy trình KTTĐ. Nó là một hệ thống tích hợp thiết lập các quy tắc tự động hóa một sản phẩm cụ thể, cụ thể là tích hợp các thư viện chức năng, các nguồn dữ liệu kiểm thử, chi tiết các đối tượng và các mô-đun có thể tái sử dụng khác nhau.

1.3.2. Chức năng của framework

Việc thiết kế các ca kiểm thử và framework là công việc hoàn toàn riêng biệt. Framework là một môi trường thực thi kiểm thử tự động, là một hệ thống tổng thể, nơi mà các trường hợp kiểm thử được thực hiện một cách tự động.

Một framework kiểm thử tự động tốt cung cấp các cấu trúc cho việc đăng nhập, báo cáo lỗi và kết hợp khả năng phục hồi. Framewok sẽ chứa các thư viện chức năng có thể được tái sử dụng trong nhiều trường hợp giống một hệ thông phần mềm bất kỳ. Và quan trọng nhất là tất cả các cấu trúc của nó sẽ giảm thiểu tối đa chi phí bảo trì mã nguồn và cung cấn một nền tảng để tạo ra các cấu trúc có thể kiểm thử được. Đây chính là điểm khởi đầu cho sự thành công trong kiểm thử tự động.

1.3.3. Mục tiêu, nhiệm vụ của framework

Framework cung cấp các cơ sở của kiểm thử tự động và đơn giản hóa các nỗ lực tự động hóa. Ưu điểm chính của một framework như vậy là chi phí bảo trì thấp. Nếu có sự thay đổi nào cho các ca

kiểm thử, thì chỉ có tệp chương trình kiểm thử của nó cần được cập nhật và các chương trình còn lại sẽ vẫn như cũ. Bên cạnh đó, ta cũng không cần phải cập nhật các kịch bản này trong trường hợp thay đổi ứng dụng.

Một framework KTTĐ có nhiệm vụ chính trong việc định nghĩa các khuôn mẫu diễn đạt được đúng yêu cầu mong đợi, tạo ra một cơ chế kiểm tra, kiểm thử phần mềm ứng dụng, thực thi việc kiểm thử và thông báo kết quả kiểm thử.

Một mục tiêu quan trọng của framework là đưa việc tạo các ca kiểm thử tách biệt khỏi với ngôn ngữ của công cụ kiểm thử và đưa đến một mức độ trừu tượng cao hơn.

1.3.4. Phân loại framework kiểm thử tự động

Có nhiều kiểu framework được cung cấp để hỗ trợ cho việc kiểm thử tự động. Việc lựa chọn các framework phù hợp, chính xác với ứng dụng sẽ giúp duy trì chi phí về nguồn nhân lực cũng như về việc bảo trì cho các kịch bản kiểm thử. Cách tiếp cận các kịch bản được sử dụng trong quá trình thử nghiệm tự động cũng ảnh hưởng đến các framework khác nhau. Các framework được phân loại theo năm mức chủ yếu: mức tuyến tính (linear), mức cấu trúc (structured), mức hướng dữ liệu (dataDriven), mức hướng từ khóa (keywordDriven) và mức hybrid (hai hoặc nhiều hơn các mô hình trên được sử dụng).

CHƯƠNG 2
GIẢI PHÁP KIỂM THỬ TỰ ĐỘNG VỚI RELEVANTCODES VÀ QUICKTESTPRO

Chương 2 của luận văn sẽ tập trung nghiên cứu về giải pháp KTTĐ với việc kết hợp sử dụng framework RelevantCodes (RC) và công cụ kiểm thử QuickTestPro (QTP). Theo đó, chương 2 sẽ trình bày một cách sơ lược về RC và QTP, sau đó trình bày sự kết hợp RC và QTP cho các ứng dụng KTTĐ.

2.1. Công cụ kiểm thử HP QuickTest Professional

HP QuickTest Professional là một công cụ KTTĐ được thiết kế để kiểm thử nhiều ứng dụng và môi trường phần mềm khác nhau, hỗ trợ kiểm thử chức năng và kiểm thử hồi quy thông qua giao diện người sử dụng. QTP hoạt động bằng cách xác định các đối tượng trên giao diện của ứng dụng hoặc của một trang web và thực hiện các hoạt động mong muốn (chẳng hạn như kích chuột hoặc các sự kiện bàn phím), nó cũng có thể bắt được các thuộc tính của đối tượng như tên hoặc ID của đối tượng.

Trong phần này sẽ trình bày một số tính năng chính của QTP và các nhược điểm của công cụ này.

2.1.1. Một số tính năng chính của QTP

QTP là công cụ KTTĐ có nhiều tính năng ưu việt nổi trội như: quản lý các điều kiện kiểm thử, quản lý xử lý ngoại lệ, cho phép kiểm thử hướng dữ liệu, có khả năng mở rộng cao, quản lý và cho

phép tùy chính các báo cáo kiểm thử, tích hợp QTP với HP Quality Center hỗ trợ quản lý quy trình KTTĐ.

2.1.2. Một số nhược điểm của QTP

QTP không thể nhận ra các đối tượng đã được người dùng tùy chỉnh và một số đối tượng phức tạp khác. Người dùng có thể định nghĩa các kiểu đối tượng này như là đối tượng ảo, tuy nhiên QTP không hỗ trợ việc ghi lại các đối tượng ảo này.

QTP chỉ chạy trong môi trường Windows, nó dựa trên phần lớn các công nghệ lạc hậu, như ActiveX. QTP cũng không thể hỗ trợ tất cả các kiểu và các phiên bản khác nhau của trình duyệt web, đặc biệt nó không hỗ trợ Safari hay Opera.

Vì chi phí bản quyền của QTP khá cáo nên công cụ này không được sử dụng rộng rãi. Phiên bản đầu tiên được phát hành là 5.5 vào năm 2001, đến năm 2010, phiên bản 11.0 đã được phát hành.

2.2. Framework RelevantCodes

Phần này trình bày cụ thể về hiện trạng của framework RelevantCodes (RC), bao gồm việc xây dựng ý tưởng, thư viện các chức năng hiện có của RC, và cấu trúc của framework RC phiên bản đầu tiên, phân tích ưu nhược điểm và cuối cùng sẽ trình bày về sự kết hợp giữa QTP và RC để thực hiện kiểm thử tự động.

2.2.1. Ý tưởng của framework RC

Có nhiều yêu cầu cho việc lựa chọn phương pháp tiếp cận công nghệ phần mềm cho framework KTTĐ. Sao cho phương pháp đó

phải hợp lý, có cơ sở để cấu trúc của kịch bản kiểm thử ổn định, không dễ dàng gặp trục trặc. Framework phải dễ dàng sử dụng, thống nhất trong cách tạo và thực thi các ca kiểm thử, có khả năng khôi phục từ những điều kiện không mong đợi. Cấu trúc mã nguồn của framework phải được kiểm thử và có khả năng bảo trì cao, có khả năng vận hành các ca kiểm thử trên nhiều máy tính cùng một lúc.

Vậy làm thể nào để đo lường được khả năng bảo trì của chương trình kiểm thử tự động, đồng thời đánh giá được phương pháp tiếp cận framework hiệu quả? RC sử dụng tỉ lệ giữa số lượng các đối tượng trên giao diện so với số lần chúng được tham chiếu đến, gọi là tỉ số 'UIObject-to-Reference', viết tắt là UtR.

2.2.2. Thư viện chức năng của RC

Thư viện các chức năng chính của RC gồm có bốn lớp: lớp Engine – đóng vai trò là trình điều khiển RC, lớp Data – trình trích xuất dữ liệu, lớp Implement – trình thực thi, lớp Support – trình hỗ trợ.

2.2.3. Cấu trúc framework của RC

Trong mục này sẽ trình bày cấu trúc tổ chức thư mục và cấu trúc tố chức của các test script cũng như cách thực thi các script.

2.2.4. Ưu nhược điểm của RC

RC có ưu điểm đầu tiên đó là mã nguồn mở, dễ phát triển. Bên cạnh đó, RC hỗ trợ phân rã các chức năng, nâng cao việc tái sử dụng mã nguồn và test script, giảm sự lặp lại không cần thiết của một số

bước trong mỗi ca kiểm thử. Ngoài ra, RC còn cho phép tùy chỉnh các thông điệp để miêu tả nhiệm vụ cho từng bước kiểm thử trong test script, hỗ trợ cho việc phân tích hiệu quả trên các báo cáo.

Song song với những ưu điểm kể trên, framework này cũng có một số nhược điểm cần được phát triển thêm. RC chưa hỗ trợ kiểm thử data-driven, không hỗ trợ kiểm tra việc lưu trữ dữ liệu trong cơ sở dữ liệu, chưa thực thi kiểm thử với nhiều tập dữ liệu khác nhau. Trong mã nguồn của phiên bản đầu tiên, RC không thể nhận dạng một số thuộc tính của sự kiện di chuyển của con trỏ chuột hay các đối tượng ‘tooltip’.

2.3. Kết hợp QTP và RC

RC làm việc trên QTP, xem QPT như là trình điều khiển cho chính nó. RC đã được thử nghiệm thành công trên QTP phiên bản 9.2, 9.5 và 10.0. QTP sử dụng một kho lưu trữ đối tượng chia sẻ tất cả các đối tượng của AUT, có thể thêm hay tái sử dụng trên các kịch bản, các tester có thể sử dụng chức năng ‘Object Spy’ của QTP để xem các thuộc tính của đối tượng, từ đó sử dụng chúng để định nghĩa và xác định đối tượng trong các ca kiểm thử, các test script vẫn được thi hành dù cho kho tự điển đối tượng trống rỗng.

QTP là một môi trường phát tiển với trình biên dich riêng, nhưng có trình soạn thảo và trình gỡ lỗi tương tự như các môi trường phát triển tích hợp khác.

2.3.1. Quy trình kiểm thử kết hợp giữa QTP và RC

Phần này mô tả các quy trình làm việc kết hợp giữa RC và QTP. Trong phần tổng quan về KTTĐ của chương 1, tôi có đề cập đến sáu pha chính của phương pháp ATLM – phương pháp luận vòng đời KTTĐ. Tôi sẽ áp dụng ý tưởng của phương pháp ATLM này để trình bày sáu giai đoạn chính của quy trình kết hợp giữa QTP và RC.

2.3.2. Mô phỏng quy trình làm việc kết hợp của RC và QTP

Để mô phỏng quy trình làm việc kết hợp của RC và QTP, sau đây tôi đưa ra một ca kiểm thử đơn giản, sau đó thiết kế, và thực thi cụ thể.

CHƯƠNG 3

CẢI TIẾN RELEVANTCODES VÀ ỨNG DỤNG VÀO KIỂM THỬ TỰ ĐỘNG

Từ việc phân tích những ưu nhược điểm của QTP và RC trong chương trước, chương này sẽ cải tiến RC bằng cách phát triển thêm một số chức năng bổ sung vào thư viện có sẵn, tổ chức lại cấu trúc framework để thuận tiện cho việc quản lý các công việc của nhóm kiểm thử. Theo đó, tôi đặt tên framework RC cải tiến là RC+.

Cấu trúc framework của RC+ sẽ được tổ chức lại để thuận tiện và dễ dàng hơn trong việc quản lý các chương trình KTTĐ. RC+ sẽ phát triển thêm thư viện các chức năng để hỗ trợ kiểm tra việc lưu trữ dữ liệu trong cơ sở dữ liệu, và thực thi ca kiểm thử với nhiều tập dữ liệu khác nhau từ một bảng dữ liệu độc lập bên ngoài.

Bên cạnh đó, RC+ cũng tùy chỉnh sự hiển thị các thông điệp miêu tả nhiệm vụ cho từng bước kiểm thử trong test script, hỗ trợ cho việc phân tích hiệu quả trên các báo cáo, cho phép tùy chỉnh hiển thị tùy ý ở bất cứ chức năng nào, ngay cả trong việc xác minh các điều kiện kiểm thử.

3.1. Hướng giải quyết và triển khai cải tiến trong RC+

Phần này sẽ trình bày hướng giải quyết cụ thể cho từng vấn đề đã đặt ra bao gồm việc cải tiến cấu trúc của chương trình KTTĐ trong tệp Data.xls, bổ sung chức năng kiểm thử cơ sở dữ liệu, bổ sung chức năng hướng dữ liệu cho RC+, hỗ trợ tùy chỉnh nhận dạng

một số đối tượng đặc biệt và cuối cùng là tùy chính hiển thị báo cáo KTTĐ.

3.1.1. Cải tiến thư viện chức năng của RC+

Cấu trúc thư viện chức năng của RC phiên bản đầu tiên có bốn lớp Engine, Data, Implement và Support. Trong RC+, bên cạnh bốn lớp này, cấu trúc thư viện chức năng được bổ sung thêm năm lớp có tên: Constant, Class Template, ExcelEngine, Filters và ReporterManager. Theo đó, cấu trúc bên trong của bốn lớp mặc định cũng thay đổi để phù hợp với cấu trúc tổng thể của thư viện chức năng mới trong RC+.

3.1.2. Cải tiến chức năng kiểm thử cơ sở dữ liệu

Trong lớp Implement của RC+, bổ sung hàm ‘CheckData’ để truy xuất đến dữ liệu tại một bảng dữ liệu trong cơ sở dữ liệu. Hàm ‘CheckData’ thực hiện kết nối đến server nơi đặt cơ sở dữ liệu của AUT, và sử dụng câu truy vấn SQL bình thường để kiểm tra giá trị trong cơ sở dữ liệu tương ứng.

3.1.3. Hỗ trợ nhận dạng các một số đối tượng đặc biệt

Bổ sung các hàm ‘CheckCellData’ để kiểm tra giá trị tại một vị trí trong đối tượng bảng, hàm ‘SelectNextRow’ để lựa chọn dòng dữ liệu kế tiếp trong đối tượng bảng, hàm ‘CheckTooltipAuditIcon’ để kiểm tra nội dung của đối tượng tooltip, hàm ‘Clear’ để xóa nội dung của đối tượng webedit, hàm ‘ClickLogout’ để thoát khỏi session cũ hoặc cookie mà trình duyệt tự động lưu lại, và tùy chỉnh báo cáo kết quả KTTĐ

3.2. Cấu trúc framework của RC+

3.2.1. Cấu trúc thư mục

Cải tiến RC+, về căn bản cấu trúc thư mục không có thay đổi nhiều, ngoại trừ việc thêm vào một số lớp thư viện chức năng.

3.2.2. Cấu trúc chương trình kiểm thử

Cải tiến RC+, cấu trúc chương trình kiểm thử được tốt chức lại, mỗi ca kiểm thử được trình bày trong một tệp tin excel riêng biệt, sau đó có một tập tin khác quản lý tất cả các ca kiểm thử. Khác xa với RC phiên bản đầu tiên, tất cả các ca kiểm thử được lưu trữ trên cùng một tập tin 'Data.xls'. RC+ sẽ có một tệp tin khác riêng biệt để quản lý tất cả các ca kiểm thử, tập tin này có tên là 'MasterData'

3.3. Thử nghiệm

Trong phần này, tôi sẽ thử nghiệm để mô phỏng quy trình kết hợp giữa framework cải tiến RC+ và QTP, đồng thời đề cập một số chức năng mà tôi đã phát triển cho RC+, cụ thể là chức năng kiểm thử cơ sở dữ liệu, hỗ trợ nhận dạng các một số đối tượng đặc biệt bao gồm đối tượng bảng, tooltip, xóa giá trị cho webedit, xóa session hay cookie đăng nhập.

3.3.1. Mô phỏng sự kết hợp của RC+ và QTP

Trong phần này, tôi sẽ sử dụng lại kịch bản đã trình bày đối với RC phiên bản đầu tiên để thử nghiệm với RC+ để dễ dàng nhận thấy sự khác biệt giữa hai framework trước và sau khi cải tiến trong việc kết hợp QPT với RC và RC+.

Sau khi thực thi xong chương trình kiểm, QPT sẽ tự động dưa ra một báo cáo chi tiết cho từng bước của chương trình kiểm thử, báo cáo có tên ‘HP Run Results Viewer’.

RC+ cũng hỗ trợ xuất ra báo cáo dưới dạng tập tin excel, bố cục tập tin excel sử dụng hai bảng tính, một bảng tính lưu thông tin tổng quát cho tất cả các chương trình kiểm thử sau một lần thực thi, bảng tính còn lại chứa thông tin chi tiết cho từng bước của ca kiểm thử.

3.3.2. Kiểm thử cơ sở dữ liệu

Sử dụng thủ tục ‘CheckData’ để truy xuất đến dữ liệu tại một bảng dữ liệu trong cơ sở dữ liệu thông qua một câu truy vấn SQL thông thường.

3.3.3. Nhận dạng các một số đối tượng đặc biệt

Sử dụng các hàm đã xây dựng để thử nghiệm cho các trường hợp kiểm tra giá trị tại một vị trí trong đối tượng bảng, nhận dạng tooltip, xóa giá trị cho webedit, thoát khỏi session cũ hoặc cookie mà trình duyệt tự động lưu lại để đảm bảo tính đúng đắn của kịch bản kiểm thử.

3.4. Nhận xét và đánh giá RC+ so với RC

So với RC, cấu trúc của RC+ hoàn toàn thay đổi, việc tổ chức lại cấu trúc framework cho phép các tester có thể làm phần việc của mình mà không ngại đến việc sẽ ảnh hưởng đến phần việc của người khác, đồng thời cách bố trí các ca kiểm thử trên từng tập tin riêng với bố cục rõ ràng, dễ dàng sử dụng, và nâng cấp, bảo trì. Đây chính là kết quả chủ chốt mà luận văn đạt được.

RC+ đã hỗ trợ thêm việc quản lý các chương trình kiểm thử của toàn nhóm kiểm thử một cách rõ ràng, dễ dàng, và tách biệt với nội dung của chương trình kiểm thử, chứ không quản lý chỉ trên một tập tin duy nhất như RC. Do đó, việc tổng hợp các chương trình kiểm thử từ mỗi tester trong nhóm kiểm thử trở nên vô cùng đơn giản và dễ quản lý nguồn gốc phát sinh lỗi.

RC+ được phát triển thêm chức năng báo cáo ở định dạng Excel cho phép tester quản lý được kết quả kiểm thử của mình, đồng thời báo cáo cho nhóm phát triển phần mềm một cách dễ dàng và cụ thể.

Bên cạnh đó, RC+ đã được phát triển thêm thư viện các chức năng để hỗ trợ kiểm tra việc lưu trữ dữ liệu trong cơ sở dữ liệu, hỗ trợ nhận dạng các một số đối tượng đặc biệt (đối tượng bảng, tooltip, xóa giá trị cho webedit, xóa session hay cookie đăng nhập), tùy chỉnh sự hiển thị các thông điệp miêu tả nhiệm vụ cho từng bước kiểm thử, hỗ trợ cho việc phân tích hiệu quả trên các báo cáo.

KẾT LUẬN

1. Các kết quả đạt được của luận văn

Luận văn được tổ chức thành ba chương chính với các nội dung nghiên cứu xoay quanh các vấn đề liên quan đến kiểm thử tự động, các công cụ kiểm thử tự động và framework kiểm thử tự động. Luận văn đã trình bày các phương pháp tối ưu trong quy trình kiểm thử tự động cũng như sự cần thiết của framework trong kiểm thử tự động.

Trong phạm vi mục tiêu của đề tài, luận văn đã đưa ra được những điểm yếu điểm mạnh, cùng các giải pháp kiểm thử tự động với RelevantCodes và QuickTestPro, cải tiến và tổ chức lại cấu trúc framework RelevantCodes, ứng dụng thành công trong kiểm thử tự động cho ứng dụng Web HTML, hỗ trợ cho việc quản lý các công việc của nhóm kiểm thử một các thống nhất và dễ dàng.

Bên cạnh đó bổ sung một số chức năng của thư viện dùng chung như đã nêu ở phần đánh giá RC+.

Kết quả nghiên cứu của luận văn có thể được đưa vào sử dụng đối với công việc kiểm thử tự động cho các ứng dụng Web HTML.

2. Hạn chế của luận văn

Từ việc phân tích những ưu nhược điểm của QTP và RC trong chương trước, có một vấn đề liên quan đến chức năng hướng dữ liệu của RC. RC chưa hỗ trợ chức năng hướng dữ liệu và RC+ cũng vậy. Mặc dù ban đầu định hướng phát triển chức năng này trong RC+ để hỗ trợ thực thi ca kiểm thử với nhiều tập dữ liệu khác nhau từ một bảng dữ liệu độc lập bên ngoài.

Luận văn đã dành nhiều sự tập trung trong việc cải tiến cấu trúc framework nên việc phát triển chức năng cho thư viện dùng chung không nhiều.

Một hạn chế nữa của cả RC và RC+, là chúng chỉ hỗ trợ cho KTTĐ các ứng dụng web dạng HTML thông thường.

3. Hướng phát triển của luận văn

Dựa trên kết quả đạt được và hạn chế đã nêu ở trên, trong tương lai, nếu có điều kiện và cơ hội để tiếp tục phát triển RC+, tôi sẽ bổ sung thêm thư viện chức năng cho RC+, phát triển chức năng hướng dữ liệu của RC+, và nghiên cứu triển khai RC+ cho nhiều ứng dụng Web khác, như Flex chẳng hạn.

Khác với phương pháp ghi/phát - mỗi ca kiểm thử là một chuỗi dài các hành động được ghi lại với bộ dữ liệu cứng, trong trường hợp muốn làm một ca kiểm thử khác, thì phải ghi một kịch bản khác để duy trì giao diện của hệ thông theo thời gian - phương pháp hướng dữ liệu sử dụng dữ liệu kiểm thử được tổ chức trong các tập tin riêng biệt, các tập tin này được đọc bằng đoạn mã KTTĐ, sau đó sử dụng như đầu vào cho hệ thống cần được kiểm thử. Bộ dữ liệu có thể chứa một lượng lớn các dữ liệu khác nhau, qua đó góp phần tăng độ bao phủ của ca kiểm thử một cách đơn giản, mà trong kiểm thử thủ công không làm được. Các ca kiểm thử có thể được thêm vào mà không cần viết thêm kịch bản kiểm thử (nếu các hành động của những ca kiểm thử này giống nhau). Các chương trình kiểm thử nghiệm cũng có thể được thực hiện để kiểm tra các giá trị hiển thị trên giao diện người dùng, nó sẽ tự động kiểm tra kết quả mong đợi. Phát triển thư

viện chức năng của RC+ để kiểm soát được trình tự các dòng dữ liệu được nhập vào hệ thống, một tập tin điều khiển sẽ quy định cụ thể trình tự dữ liệu, và chuỗi các hành động giống như đối với kiểm thử thủ công.

Đối với các ứng dụng web sử dụng công nghệ Flex, theo như tìm hiểu sơ qua, có thể ta chỉ không cần sử dụng QTP để kết hợp với RC mà ta có thể sử dụng một công cụ mã nguồn mở khác, đó là Selenium, và tích hợp với RelevantCodes để phát triển.