**CHƯƠNG 1: TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM**

* 1. **KIẾM THỨC KIỂM THỬ PHẦN MỀM CƠ SỞ**
     1. Chất lượng phần mềm và đảm bảo chất lượng phần mềm
        1. Định nghĩa chất lượng phần mềm

- Có rất nhiều định nghĩa về chất lượng phần mềm được đưa ra bởi các tổ chức, cá nhân khác nhau. Trong phạm vi của bài viết này trình bày một số định nghĩa tiêu biểu.

\***Định nghĩa theo IEEE(1991):**

* **Định nghĩa 1:** Chất lượng phần mềm là một mức độ mà một hệ thống, thành phần hệ thống hay tiến trình đáp ứng được yêu cầu đã được đặc tả.
* **Định nghĩa 2:** Chất lượng phần mềm là mức độ mà một hệ thống, thành phần hệ thống hay tiến trình đáp ứng được yêu cầu và sự mong đợi của khách hàng hay người sử dụng.

\***Phân tích hai quan điểm của IEEE:**

* Theo quan điểm thứ nhất của IEEE: chúng ta sẽ bị phụ thuộc quá nhiều vào tài liệu đặc tả của yêu cầu, dẫn đến nếu việc xấc định yêu cầu bị sai, thiếu thì một phần mềm được làm đúng với đặc tả chưa chắc đã là một phần mềm có chất lượng.
* Theo quan điểm thứ hai của IEEE: khách hàng đôi khi không có kiến thức về công nghệ, họ có thể đưa ra những mong muốn hết sức vô lý và có thể thay đổi yêu cầu với phần mềm nhiều lần, thậm chí thay đổi ngay trong giai đoạn cuối. Điều này gây nhiều khó khăn cho việc phát triển phần mềm.

\***Định nghĩa theo Pressman:** Chất lượng phần mềm là sự phù hợp của các yêu cầu cụ thể về hiệu năng và chức năng, các tiêu chuẩn phát triển phần mềm được ghi lại rõ ràng bằng tài liệu với các đặc tính ngầm định của tất cả các phần mềm được phát triển chuyên nghiệp.

Định nghĩa của Pressman đề xuất ba yêu cầu với chất lượng phần mềm phải được đáp ứng khi phát triển phần mềm:

* Các yêu cầu chức năng rõ ràng là nhân tố chính quyết định chất lượng đầu ra của phần mềm.
* Các tiêu chuẩn chất lượng phần mềm sẽ được nói đến trong hợp đồng.
* Các đặc tính ngầm định cần được đáp ứng trong quá trình phát triển cho dù không được nói đến rõ ràng trong hợp đồng.
  + - 1. Định nghĩa đảm bảo chất lượng phần mềm

**- Định nghĩa theo Daniel Galin**: Đảm bảo chất lượng phần mềm (Software Quality Assure) 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.

* + 1. Lỗi phần mềm
       1. Định nghĩa lỗi phần mềm và phân loại lỗi phần mềm
* Định nghĩa lỗi phần mềm: Có rất nhiều định nghĩa về lỗi phần mềm nhưng có thể hiểu và phát biểu một cách đơn giản thì "Lỗi phần mềm là sự không khớp giữa chương trình và đặc tả của nó".
* Dựa vào định nghĩa, ta có thể phân loại lỗi phần mềm thành 3 dạng:
  + Lỗi sai: Sản phẩm phần mềm được xây dựng khác với đặc tả.
  + Lỗi thiếu: Các yêu cầu của sản phẩm phần mềm đã có trong đặc tả nhưng lại không có trong sản phẩm thực tế.
  + Lỗi thừa: Sản phẩm thực tế có những tính năng không có trong tài liệu đặc tả.
    - 1. Các nguyên nhân gây lỗi phần mềm

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

\***Định nghĩa các yêu cầu bị lỗi:** Những lỗi trong việc xác định yêu cầu thường nằm ở phía khách hàng. Một số lỗi thường gặp là: định nghĩa sai yêu cầu, lỗi không hoàn chỉnh, thiếu các yêu cầu quan trọng hoặc là quá chú trọng các yêu cầu không thật sự cần thiết.

\***Các lỗi trong giao tiếp giữa khách hàng và nhà phát triển:** Hiểu lầm trong giao tiếp giữa khách hàng và nhà phát triển cũng là nguyên nhân gây lỗi. Những lỗi này thường xuất hiện trong giai đoạn đầu của dự án. Một số lỗi hay gặp phải: hiểu sai chỉ dẫn trong tài liệu yêu cầu, hiểu sai thay đổi khi khách hàng trình bày bằng lời nói và tài liệu, hiểu sai về phản hồi và thiếu quan tâm đến những đề cập của khách hàng.

* Giải pháp khắc phục: Cần có ủy ban liên kết giữa khách hàng và nhà cung cấp để tránh lỗi trong giao tiếp. Ủy ban do quản trị dự án đứng đầu và khách hàng phải giới thiệu những người hiểu về mặt nghiệp vụ vào ủy ban đó.

\***Sai lệch có chủ ý với các yêu cầu phần mềm:** Trong một số trường hợp các nhà phát triển cố tình làm sai lệnh các yêu cầu trong tài liệu đặc tả. Nguyên nhân của việc này đến từ các áp lực thời gian, ngân sách, hay cố tình sử dụng lại các mô-đun từ các dự án trước mà chưa phân tích đầy đủ những thay đổi để thích nghi với các yêu cầu mới.

* Giải pháp khắc phục: Dựa trên những thống kê để quyết định xem giải pháp như thế nào, sắp xếp ưu tiên xem bỏ được yêu cầu nào hay sử dụng lại được mô-đun nào.

\***Các lỗi thiết kế logic:** Lỗi phần mềm xảy ra trong quá trình các chuyên gia thiết kế hệ thống, các kiến trúc sư hệ thống, kỹ sư phần mềm, các nhà phân tích xây dựng phần mềm theo yêu cầu. Các lỗi điển hình bao gồm:

* Định nghĩa các yêu cầu phần mềm bằng các thuật toán sai.
* Quy trình định nghĩa có chứa trình tự lỗi.
* Sai sót trong các định nghĩa biên như > 3 hay ≥ 3.
* Thiếu sót các trạng thái hệ thống phần mềm được yêu cầu.

\***Các lỗi lập trình:** Có rất nhiều lý do dẫn đến việc các lập trình viên gây ra các lỗi lập trình. Những lý do này bao gồm: sự hiểu sai các tài liệu thiết kế, ngôn ngữ; sai sót trong ngôn ngữ lập trình; sai sót trong việc áp dụng các công cụ phát triển; sai sót trong lựa chọn dữ liệu...

\***Không tuân thủ theo các tài liệu hướng dẫn và tiêu chuẩn lập trình:** Các lỗi phần mềm có thể đến từ việc không tuân thủ các tài liệu và tiêu chuẩn lập trình của các tổ chức phát triển phần mềm.

\***Thiếu sót trong quá trình kiểm thử:** Lỗi phần mềm có thể đến từ chính quá trình kiểm thử khi mà người kiểm thử để lọt lỗi.

Những lỗi này đến từ các nguyên nhân sau đây:

* Kế hoạch kiểm thử chưa hoàn chỉnh, để sót yêu cầu cần kiểm thử.
* Lỗi trong tài liệu và báo cáo kiểm thử.
* Việc sửa chữa các lỗi được phát hiện không hoàn chỉnh do áp lực thời gian hay do thiếu cẩn thận.

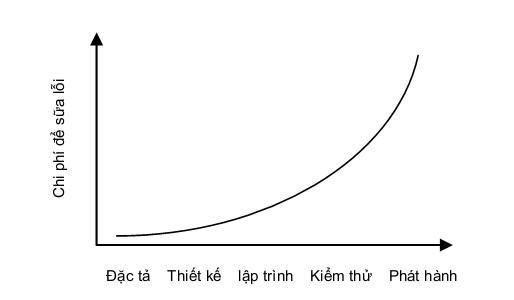
Giải pháp: Lên kế hoạch kiểm thử cụ thể tại giai đoạn đầu của dự án.

\***Các lỗi thủ tục:** Các thủ tục hướng dẫn cho người sử dụng tại từng bước của tiến trình. Chúng có tầm quan trọng đặc biệt trong các hệ thống phần mềm phức tạp mà các tiến trình được thực bằng nhiều bước, mỗi bước có thể có nhiều kiểu dữ liệu và cho phép kiểm tra các kết quả trung gian. Các lỗi có thể đến từ việc viết các thủ tục.

\***Các lỗi về tài liệu:** Các lỗi về tài liệu là vấn đề của các đội phát triển và bảo trì khi có những sai sót trong các tài liệu liên quan. Những lỗi này có thể là nguyên nhân gây ra lỗi trong giai đoạn phát triển kế tiếp và giai đoạn bảo trì.

* + - 1. Chi phí cho việc sửa lỗi phần mềm

- Việc kiểm thử và sửa lỗi phần mềm có thể thực hiện trong bất cứ giai đoạn nào của vòng đời phần mềm. Tuy nhiên công việc này nên được thực hiện càng sớm càng tốt vì càng về giai đoạn sau của vòng đời phần mềm, chi phí cho việc tìm và sửa lỗi càng tăng, đặc biệt là đến giai đoạn đã triển khai phần mềm thì chi phí cho sửa lỗi sẽ trở nên rất lớn và ảnh hưởng trực tiếp đến uy tín của tổ chức phát triển phần mềm.

Theo tài liệu của Boehm, chi phí cho việc tìm và sửa lỗi phần mềm sẽ tăng theo hàm mũ trong biểu đồ sau:

\_Sơ đồ 1: Chi phí cho việc sửa lỗi phần mềm \_

* + 1. Khái niệm kiểm thử phần mềm
* Theo Glenford Myers: Kiểm thử là quá trình vận hành chương trình để tìm ra lỗi
* TheoIEEE: Kiểm thử là:

-(1)Là quá trình vận hành hệ thống hoặc thành phần dưới những điều kiện xác định, quan sát hoặc ghi nhận kết quả đánh giá về hệ thống hoặc thành phần đó.

-(2)Là quá trình phân tích phần mềm để tìm ra sự khác biệt giữa điều kiện thực tế và điều kiện yêu cầu và dựa vào điểm khác biệt đó để đánh giá tính năng phần mềm.

* + 1. Mục tiêu của kiểm thử
* Tìm các bug phát sinh do dev tạo ra khi code.
* Đạt được sự tự tin và cung cấp thông tin về mức độ chất lượng.
* Để ngăn ngừa lỗi.
* Đảm bảo rằng kết quả cuối cùng đáp ứng các yêu cầu kinh doanh và người sử dụng.
* Để đạt được sự tín nhiệm của khách hàng bằng cách cung cấp cho họ một sản phẩm chất lượng.

Kiểm thử phần mềm sẽ giúp hoàn thiện các ứng dụng phần mềm hoặc sản phẩm so với yêu cầu kinh doanh và người sử dụng. Nó là rất quan trọng để đảm bảo kiểm thử tốt để kiểm thử các ứng dụng phần mềm hoàn toàn và chắc chắn rằng nó hoạt động tốt và theo các thông số kỹ thuật.

Việc xác định phạm vi kiểm tra các trường hợp kiểm thử nên được thiết kế tốt với khả năng tối đa của việc tìm kiếm các lỗi hiệu quả và được tính toán là số bug báo cáo cho mỗi trường hợp kiểm thử.

Kiểm tra phần mềm để chắc chắn kiểm thử đang thực hiện đúng cách và hệ thống đã sẵn sàng để sử dụng. Kiểm thử bao phủ các lĩnh vực khác nhau như: chức năng của các ứng dụng, khả năng tương thích của các ứng dụng với các hệ điều hành, phần cứng và các loại khác nhau của các trình duyệt, thực hiện kiểm thử để kiểm tra hiệu năng của các ứng dụng để đảm bảo rằng hệ thống đáng tin cậy và không có trục trặc hay không nên có bất kỳ vấn đề cản trở. Xác định rằng các ứng dụng có thể được triển khai một cách dễ dàng với máy tính và không có bất kỳ sự cố. Do đó các ứng dụng rất dễ dàng để cài đặt, tìm hiểu và sử dụng.

Kiểm thử phần mềm cho phép tạo ra những đánh giá khách quan về mức độ phù hợp của hệ thống các yêu cầu đã nêu và thông số kỹ thuật.

Kiểm tra xác nhận rằng hệ thống đáp ứng các yêu cầu khác nhau bao gồm: chức năng, hiệu suất, độ tin cậy, an toàn, khả năng sử dụng và như vậy. Việc xác nhận này được thực hiện để đảm bảo rằng chúng tôi đang xây dựng hệ thống phù hợp.

Xác nhận để đảm bảo đang xây dựng hệ thống phù hợp. Ngoài việc giúp đưa ra quyết định, các thông tin từ các kiểm thử phần mềm giúp quản lý rủi ro.

* + 1. Tầm quan trọng của kiểm thử phần mềm

Người phát triển phần mềm cho rằng:

* Kiểm thử để chứng minh chương trình không có lỗi
* Mục đích của kiểm thử là chỉ ra rằng chương trình đã thực hiện đúng các chức năng đã đưa ra.
* Kiểm thử là quy trình thực hiển để chứng tỏ chương trình đã làm được các chức năng cần có
* Kiểm thử còn để tìm lỗi và sửa chữa các lỗi đó nhằm tăng thêm độ tin cậy cho phần mềm

Vai trò của kiểm thử tròn suốt quá trình sống của phần mềm:

* Kiểm thử không tồn tại độc lập
* Các hoạt động của kiểm thử luôn gắn liền với các hoạt động phát triển phần mềm
* Các mô hình phát triển phần mềm khác nhau cần các cách tiếp cận kiểm thử khác nhau
  + 1. Các nguyên tắc trong kiểm thử

Trong kiểm thử các 7 nguyên tắc cơ bản:

1, Kiểm thử chỉ ra sự hiện diện của lỗi trong phần mềm

2, Kiểm thử tất cả các trường hợp là không thể

3, Nên thực hiện kiểm thử càng sớm càng tốt

4, Sự phân cụm của các lỗi

5, Ngịch lý thuốc trừ sâu

6, Kiểm thử theo các ngữ cảnh độc lập

7, Sự sai lầm về việc không có lỗi

* + 1. Phân loại kiểm thử
* Dựa vào mục đích kiểm thử

- Kiểm thử đơn vị, module

- Kiểm thử cấu hình

- Kiểm thử sơ lược

- Kiểm thử tích hợp

- Kiểm thử chức năng

- Kiểm thử hồi quy

- Kiểm thử hệ thống

- Kiểm thử tải dữ liệu

- Kiểm thử tải trọng

- Kiểm thử hiệu suất

- Kiểm thử chấp nhận

- Kiểm thử bảo mật

* Dựa vào chiến lược kiểm thử

-Kiểm thử thủ công: Thực hiện kiểm thử mọi thứ bằng tay

-Kiểm thử tự động: Kiểm thử một cách tự động nhờ công cụ kiểm thử

* Dựa vào phương pháp tiến hành

-Kiểm thử tĩnh: thường không kiểm thử chi tiết mà kiểm tra tính đúng đắn của thuật toán, code, tài liệu

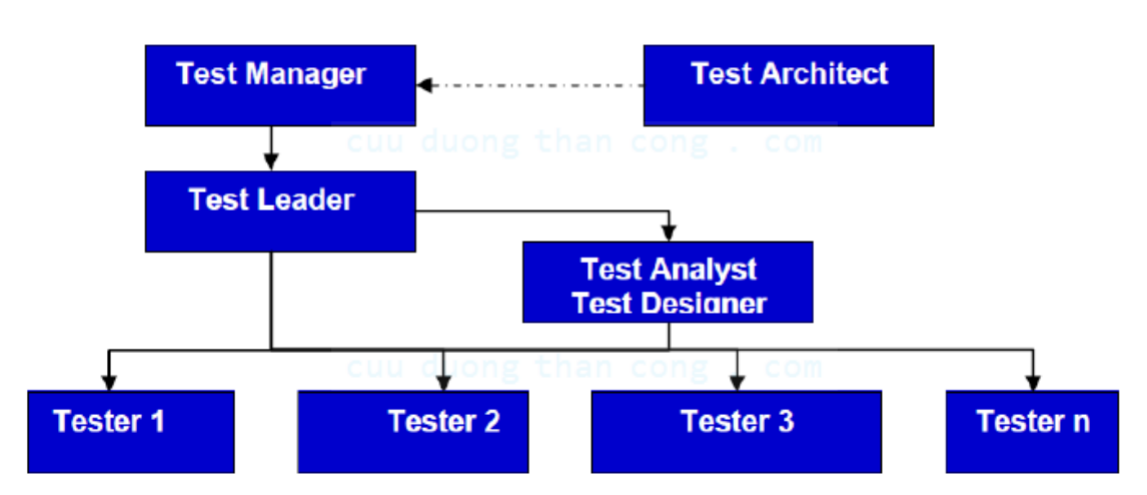
- Kiểm thử động: chạy mã lập trình thực tế trong các tình huống, diễn ra khi bản thân chương trình đó đang được sử dụng

* Dựa vào kỹ thuật kiểm thử

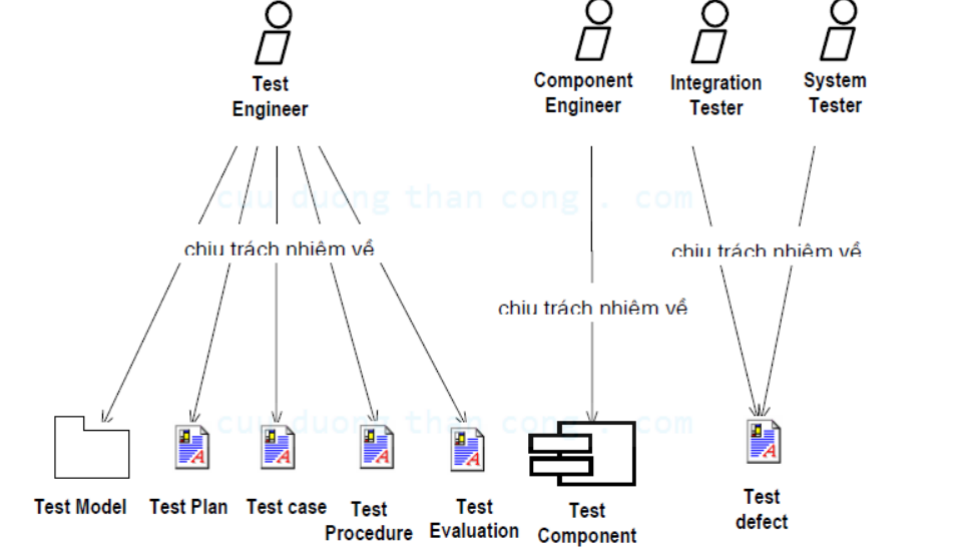
-Kiểm thử hộp trắng

-Kiểm thử hộp đen

* + 1. Đối tượng thực hiện kiểm thử



Sơ đồ tổ chức của đội kiểm thử



Các worker và quy trình

* + 1. Các hạn chế của kiểm thử
* Không thể chắc chắn đặc tả phần mềm đúng hoàn toàn
* Không thể chắc chắn hệ thống hay tool kiểm thử là đúng
* Không có tool kiểm thử nào phù hợp với mọi phần mềm
* Kỹ sư kiểm thử không thể chắc chắn hiểu hết về phần mềm
* Không có tài nguyên để kiểm thử tất cả
* Không thể tìm ra được tất cả lỗi
  1. **KỸ THUẬT KIỂM THỬ THỦ CÔNG VÀ TỰ ĐỘNG**
     1. Kỹ thuật kiểm thử thủ công (Manual testing)
        1. Khái niệm:

- Kiểm thử thủ công là kiểm thử một phần mềm một cách thủ công (không sử dụng bất kỳ công cụ tự động hoặc bất kỳ đoạn mã nào). Với loại kiểm thử này, tester như người sử dụng cuối sẽ kiểm tra phần mềm để xác định bất kỳ hành vi không mong muốn hoặc lỗi . Có rất nhiều giai đoạn để kiểm thử bằng tay như Kiểm thử đơn vị (Unit testing), Kiểm thử tích hợp (Integration testing), Kiểm thử hệ thống (System testing) và Kiểm thử chấp nhận (User Acceptance testing).

* + - 1. Mục tiêu của kiểm thử tự động

- Kiểm thử thủ công là đảm bảo rằng ứng dụng hoạt động phù hợp với các yêu cầu chức năng được chỉ định.

- Kiểm thử thủ công cũng đảm bảo rằng các lỗi đã tìm thấy được sửa chữa bởi các developer và được kiểm thử lại bởi những tester sau khi các lỗi được khắc phục.

* + - 1. Các loại kiểm thử thủ công
* Black Box Testing
* White Box Testing
* Unit Testing
* System Testing
* Integration Testing
* Acceptance Testing
  + - 1. Cáchthực hiện kiểm thử thủ công

1. Đọc và hiểu tài liệu của dự án phần mềm. Ngoài ra, nghiên cứu Ứng dụng khi thực hiện kiểm thử (AUT) nếu có.
2. Dự thảo kiểm thử bao gồm tất cả các yêu cầu được đề cập trong tài liệu.
3. Xem xét và vạch ra các trường hợp thử nghiệm với Trưởng nhóm, Khách hàng (nếu có)
4. Thực hiện các trường hợp kiểm thử trên AUT
5. Báo cáo lỗi.
6. Khi các lỗi đã được sửa sẽ được tester thực hiện một lần nữa các trường hợp kiểm thử thất bại để xác minh rằng lỗi đã được khắc phục.
   * + 1. Ưu nhược điểm

Ưu điểm:

* Nhận phản hồi trực quan nhanh và chính xác
* Ít tốn kém hơn vì không cần phải chi ngân sách cho các công cụ và quy trình tự động hóa
* Sự phán đoán và trực giác của con người luôn có lợi cho yếu tố thủ công
* Trong khi kiểm thử một thay đổi nhỏ, một kiểm thử tự động hóa sẽ yêu cầu coding có thể tốn thời gian. Trong khi kiểm thử thủ công lại không tốn thời gian.

Nhược điểm:

* Phương pháp kiểm thử thủ công ít đáng tin cậy hơn vì được thực thi bởi con người. Do đó, dễ mắc sai lầm & không tìm thấy lỗi.
* Quá trình kiểm thử thủ công không thể được ghi lại, do đó không thể sử dụng lại.
* Một số phần nhất định khó thực hiện thủ công, có thể cần thêm thời gian.
  + 1. Kỹ thuật kiểm thử tự động(Automation testing)
       1. Khái niệm:

- Kiểm thử tự động (Automation testing) là khi tester viết kịch bản và sử dụng những công cụ hỗ trợ để kiểm thử phần mềm. Quá trình này là chuỗi tự động hóa quá trình kiểm thử thủ công. Kiểm thử tự động được sử dụng để chạy lại các kịch bản kiểm thử đã được kiểm thử thủ công, nhanh chóng, lặp lại nhiều lần.

- Ngoài kiểm thử hồi quy, kiểm thử tự động cũng được sử dụng để kiểm tra ứng dụng từ load, performance, và stress. Nó tăng thêm độ bao phủ, cải thiện độ chính xác ,tiết kiệm thời gian và tiền bạc so với kiểm thử thủ công

* + - 1. Tại sao cần kiểm thử tự động

Kiểm thử tự động rất quan trọng vì những lý do sau:

* Kiểm thử thủ công tất cả các kịch bản nhiều lần sẽ gây tốn thời gian và tiền bạc
* Rất khó để kiểm thử thủ công các trang web hay phần mềm đa ngôn ngữ
* Kiểm thử tự động hóa không cần sự can thiệp của con người. Có thể chạy kiểm thử tự động qua đêm mà không cần giám sát.
* Kiểm thử tự động hóa làm tăng tốc độ thực hiện kiểm thử
* Kiểm thử tự động hóa giúp tăng phạm vi kiểm thử
* Kiểm thử thủ công có thể trở nên nhàm chán, do đó dễ bỏ sót lỗi.
  + - 1. Tự động hóa là gì ?

- Không phải tự động hóa lúc nào cũng khả thi với mọi phần mềm. Các lĩnh vực như người dùng có thể thực hiện giao dịch cũng như form đăng nhập hoặc form đăng ký, bất kỳ lĩnh vưc nào có số lượng người dùng lớn có thể truy cập đồng thời vào phần mềm thì nên được tự động hóa.

* + - 1. Tự động hóa như thế nào?

Tự động hóa được thực hiện bằng cách sử dụng một ngôn ngữ máy tính được hỗ trợ như VB và ứng dụng phần mềm tự động. Có rất nhiều công cụ có sẵn có thể được sử dụng để viết những đoạn kịch bản tự động. Trước khi nhắc đến các công cụ đó, hãy xác định kế hoạch để tự động hóa quá trình kiểm thử:

* Xác định các lĩnh vực trong phần mềm để tự động hoá.
* Lựa chọn những công cụ thích hợp cho kiểm thử tự động hóa.
* Viết mã tự động hóa (test scripts).
* Phát triển các bộ kiểm tra (test suits).
* Thực thi các đoạn mã.
* Tạo các báo cáo kết quả kiểm thu.
* Phát hiện các lỗi tiềm ẩn hoặc vấn đề hiệu suất.
  + - 1. Quy trình kiểm thử tự động

- Lựa chọn công cụ kiểm thử: Lựa chọn Công cụ kiểm thử phần lớn phụ thuộc vào công nghệ mà Ứng dụng Kiểm thử được xây dựng. Chẳng hạn, QTP không hỗ trợ Informatica. Vì vậy, QTP không thể được sử dụng để kiểm thử các ứng dụng của Informatica.

- Xác định phạm vi tự động: Phạm vi tự động là phạm vi Ứng dụng được Kiểm thử tự động hóa. Các điểm sau giúp xác định phạm vi tự động:

* Các tính năng quan trọng
* Kịch bản có lượng dữ liệu lớn
* Các chức năng chung trên ứng dụng
* Tính khả thi về mặt kỹ thuật
* Mức độ các thành phần được sử dụng lại
* Độ phức tạp của test cases
* Khả năng sử dụng các test cases tương tự để kiểm thử trên nhiều trình duyệt

- Lập kế hoạch, thiết kế và phát triển: Tester tạo một chiến lược & kế hoạch kiểm thử tự động, chi tiết như sau:

* Công cụ kiểm thử tự động được chọn
* Thiết kế Framework và các tính năng của Framework
* Các mục trong phạm vi và ngoài phạm vi tự động hóa
* Chuẩn bị kiểm thử tự động hóa
* Lịch trình và mốc thời gian thực thi kịch bản kiểm thử
* Bàn giao sản phẩm được kiểm thử tự động

- Thực thi kiểm thử: Kịch bản kiểm thử tự động hóa được thực thi trong giai đoạn này. Các Scripts cần dữ liệu kiểm thử đầu vào trước khi được thiết lập để chạy. Sau khi thực hiện sẽ trả về các báo cáo kiểm thử chi tiết.

- Bảo trì: Khi các chức năng mới được thêm vào Hệ thống được kiểm thử với các chu kỳ liên tiếp, Scripts tự động hóa cũng cần được thêm, xem xét và bảo trì cho mỗi chu kỳ phát hành. Bảo trì trở nên cần thiết để cải thiện hiệu quả của Scripts trong kiểm thử tự động hóa.

- Các công cụ kiểm thử tự động hóa

* Tricentis
* Mabl
* Selenium
* QTP(MicroF Focus UFT)
* Rational Functional Tester
* Witir
* Skill test

- Ưu nhược điểm

Ưu điểm:

* Vì hầu hết các phần của quy trình kiểm thử là tự động, bạn có thể có một quy trình nhanh chóng và hiệu quả
* Quá trình tự động có thể được ghi lại. Điều này cho phép bạn sử dụng lại và thực hiện cùng loại hoạt động kiểm thử
* Kiểm thử tự động được thực hiện bằng các công cụ phần mềm, do đó, có thể hoạt động liên tục mà không biết mệt mỏi.
* Có thể dễ dàng tăng năng suất, cung cấp kết quả kiểm thử nhanh & chính xác
* Kiểm thử tự động giúp bạn tìm thấy nhiều lỗi hơn so với tester

Nhược điểm:

* Không có yếu tố con người, thật khó để hiểu sâu hơn về các khía cạnh trực quan của giao diện người dùng như màu sắc, phông chữ, kích thước, độ tương phản hoặc kích thước nút bấm, layout...
* Các công cụ để chạy kiểm thử tự động có thể tốn kém, có thể làm tăng chi phí kiểm thử của dự án.
* Công cụ kiểm thử tự động chưa phải là bằng chứng đầy đủ. Mỗi công cụ tự động hóa đều có những hạn chế của chúng làm, giảm phạm vi tự động hóa.
  1. **MỘT SỐ CÔNG CỤ KIỂM THỬ TỰ ĐỘNG**

**Các công cụ kiểm thử của Selenium: Selenium là một bộ kiểm thử tự động mã nguồn mở miễn phí cho các ứng dụng web trên các trình duyệt và nền tảng khác nhau.** Nó khá giống với HP Quick Test Pro (QTP) nhưng Selenium chỉ tập trung vào việc tự động hóa các ứng dụng dựa trên web. Kiểm thử được thực hiện bằng công cụ Selenium thường được gọi là Selenium Testing.

Trong Selenium có 4 phần:

1. Selenium IDE

2. Selenium RC

3. Selenium WebDriver

4. Selenium Grid

* + 1. Selenium IDE

Môi trường phát triển tích hợp Selenium (IDE) là framework đơn giản nhất trong bộ Selenium. Nó là một plugin Firefox mà bạn có thể cài đặt dễ dàng như các plugin khác. Tuy nhiên, vì sự đơn giản của nó, nên chỉ sử dụng Selenium IDE làm công cụ tạo mẫu. Nếu bạn muốn tạo thêm các trường hợp kiểm thử nâng cao, bạn sẽ cần phải sử dụng hoặc là Selenium RC hoặc WebDriver.

* + 1. Selenium RC

Selenium RC là **framework kiểm thử hàng đầu** của toàn bộ dự án Selenium trong một thời gian dài. Đây là công cụ kiểm thử web tự động đầu tiên**cho phép người dùng sử dụng đa dạng ngôn ngữ lập trình để làm việc với nó.** Kể từ phiên bản 2.25.0, RC có thể hỗ trợ các ngôn ngữ lập trình sau: Java, C#, PHP, Python, Perl, Ruby

* + 1. Selenium WebDriver

WebDriver tốt hơn cả Selenium IDE và Selenium RC ở nhiều khía cạnh. Nó là một cách tiếp cận hiện đại và ổn định hơn trong việc tự động hóa các hành động của trình duyệt. WebDriver, không giống như Selenium RC, không dựa vào JavaScript cho tự động hóa. Nó kiểm soát trình duyệt bằng cách giao tiếp trực tiếp với nó. Các ngôn ngữ được hỗ trợ giống như ngôn ngữ trong Selenium RC: Java, C#, PHP, Python, Perl, Ruby

* + 1. Selenium Grid

Selenium Grid là một công cụ **được sử dụng cùng với Selenium RC để chạy kiểm thử song song** trên các máy khác nhau và các trình duyệt khác nhau cùng một lúc. Thực thi song song có nghĩa là chạy nhiều kiểm thử cùng một lúc.

* + 1. Côngcụ kiểm thử Jmeter

JMeter là một phần mềm có thể được sử dụng để thực hiện kiểm tra hiệu năng (**performance testing**), kiểm tra tải (**load testing**) và kiểm tra chức năng (**functional testing**) của các ứng dụng web của bạn. JMeter cũng có thể mô phỏng tốc độ tải trên một máy chủ bằng cách tạo ra hàng ngàn người dùng ảo đồng thời truy cập máy chủ web.

* + 1. Katalon (katalon studio)

- Là một bộ công cụ toàn diện cho kiểm thử tự động hoá ứng dụng trên web và điện thoại di động. Nó có đầy đủ các tính năng mạnh mẽ để vượt qua được những vấn đề phổ biến trong tự động hoá thử nghiệm giao diện web (ví dụ: pop-up, iFrame, wait-time). Nhờ đó, tester thực hiện công việc tốt hơn, nhanh hơn, tăng cao chất lượng phần mềm qua các giải pháp thông minh, thân thiện và linh hoạt.

* + 1. Junit

- Là một framework mã nguồn mở, miễn phí, quan trọng trong sự phát triển của test – driven của lập trình viên về hướng đối tượng java.

* + 1. Appium

- Là một công cụ kiểm thử tự động mã nguồn mở để tự động hoá các ứng dụng di động native và hybrid. Nó được biết đến như một công cụ kiểm thử

* + 1. Côngcụ kiểm thử LoadStorm

- LoadStorm là một công cụ kiểm thử cho các ứng dụng web và mobile. LoadStorm là một công cụ kiểm tra tải SaaS. Nó thử nghiệm hiệu suất theo yêu cầu, kiểm tra tải và thử nghiệm ứng suất cho các ứng dụng web và trang web. Nó tạo giúp cho việc tìm kiếm các dữ liệu có vấn đề của trang web của bạn bằng cách cung cấp báo cáo phân tích sâu rộng trên máy chủ, từng trang hoặc theo loại yêu cầu cho mọi chỉ số hiệu suất.

* + 1. Côngcụ kiểm thử QTP

- Quick Test Professional (QTP) là một công cụ kiểm thử tự động được thiết kế bởi Mercury Interactive và sau đó được mua lại bởi HP. QTP giúp tester tiến hành các kiểm tra một cách tự động để xác định errors, defects khác với kết quả mong muốn của ứng dụng, phần mềm hay chức năng... mà ta đang kiểm tra. QTP được dùng để kiểm thử chức năng (functional test) và cho phép thực hiện kiểm thử hồi qui (regression test) một cách tự động

* + 1. Côngcụ kiểm thử Robotium

- Robotium là một công cụ viết bằng mã nguồn mở dùng để kiểm thử hộp xám tự động dành cho các ứng dụng trên điện thoại hệ điều hành android. Robotium giúp viết các test case về test chức năng, hệ thống, và acceptance test scenarios, bao phủ các tính năng của Android.

* + 1. Côngcụ kiểm thử SOASTA CloudTest

- SOASTA CloudTest hỗ trợ chức năng quản lý người dùng (RUM - Real User Monitoring) và cho phép những người dùng xây dựng những bài test sử dụng những dữ liệu dựa trên cơ chế quản lý người dùng (RUM based data) và những kịch bản ứng dụng cụ thể. SOASTA CloudTest có thể kiểm tra khả năng chịu tải của các ứng dụng theo vị trí địa lý khác nhau, đặc biệt 2 khâu integration và phân tích thời gian thực giữa các monitoring, test design, reporting đều được tiến hành một cách liền mạch.