[**NGUYÊN TẮC CƠ BẢN CỦA KIỂM THỬ** 2](#_Toc437379795)

[**1.** **Tại sao kiểm thử là cần thiết.** 2](#_Toc437379796)

[**1.1** **Giới thiệu.** 2](#_Toc437379797)

[**1.2** **Kiểm thử là gì?** 2](#_Toc437379798)

[**1.3** **Bối cảnh của các phần mềm.** 4](#_Toc437379799)

[**1.4** **Nguyên nhân của các lỗi phần mềm.** 4](#_Toc437379800)

[**1.5** **Chi phí cho việc sửa lỗi phần mềm.** 5](#_Toc437379801)

[**1.6** **Vai trò của kiểm thử trong phát triển, bảo trì và hoạt động của phần mềm.** 6](#_Toc437379802)

[**1.7** **Kiểm thử và chất lượng.** 6](#_Toc437379803)

[**1.8** **Kiểm thử bao nhiêu là đủ?** 8](#_Toc437379804)

[**2.** **Các nguyên tắc kiểm thử.** 9](#_Toc437379805)

[**3.** **Quy trình kiểm thử cơ bản.** 10](#_Toc437379806)

[**3.1** **Kế hoạch kiểm thử và sử lý.** 10](#_Toc437379807)

[**3.2** **Phân tích và thiết kế kiểm thử.** 10](#_Toc437379808)

[**3.3** **Thực thi kiểm thử.** 11](#_Toc437379809)

[**3.4** **Đánh giá tiêu chí và thực hiện báo cáo.** 12](#_Toc437379810)

[**3.5** **Những hoạt động chấm dứt quá trình kiểm thử.** 12](#_Toc437379811)

[**4.** **Tâm lý học của kiểm thử.** 12](#_Toc437379812)

[**5.** **Mức độ ưu tiên của kiểm thử.** 13](#_Toc437379813)

[**5.1** **Các tiêu chí xếp hạng.** 13](#_Toc437379814)

[**5.2** **Độ nghiêm trọng.** 14](#_Toc437379815)

[**5.3** **Độ ưu tiên.** 15](#_Toc437379816)

**NGUYÊN TẮC CƠ BẢN CỦA KIỂM THỬ**

1. **Tại sao kiểm thử là cần thiết.**
   1. **Giới thiệu.**

Trong tài liệu này chúng ta sẽ đi tìm hiểu tại sao kiểm thử lại cần thiết bằng cách đi mô tả và minh họa làm thế nào mà những lỗi phần mềm ảnh hưởng đến con người. Chúng ta cũng sẽ đi giải thích tại sao kiểm thử là cần thiết. Làm thế nào để kiểm thử một phần mềm, cách kiểm thử để đảm bảo chất lượng của phần mềm và một số nguyên tắc cơ bản của kiểm thử.

Xuyên suốt tài liệu này chúng ta sẽ biết về các thuật ngữ bugs, defect, error, failure, fault, mistake, quality, risk, testing, software, testing và exhaustive testing. Bạn sẽ tìm thấy được những thuật ngữ này trong bảng chú giải.

Để bắt đầu chúng ta hãy xem xét tại sao kiêm thử là cần thiết. Kiểm thử là cần thiết bởi vì mọi người đều mắc những sai lầm. Một vài lỗi thì không quan trọng, nhưng một vài lỗi thì thực sự đắt giá và đáng nguy hiểm. Chúng ta cần kiểm tra mọi thứ và bất kì thứ gì chúng ta tạo ra bởi vì mọi thứ đều có thể có những sai xót. “Humans make the mistakesall the time”.

Chúng ta nên giả định rằng công việc của chúng ta chứa những lỗi, chúng ta cần phải kiểm tra lại nó. Tuy nhiên một vài lỗi đến từ các giả định không đúng, hoặc các điểm mù, do vây chúng ta thường mắc những lỗi giống nhau khi chúng ta đi kiểm tra lại chính công việc của mình và thường thì chúng ta không tìm ra được những lỗi đấy. Do vậy chúng ta phải chú ý đến những thiếu sót này để tránh mắc những sai lầm tương tự lần sau. Một các tốt nhất đó là chúng ta hãy để một người khác kiểm tra công việc của mình, một người có khả năng để phát hiện ra các lỗi mà chúng ta mắc phải.

* 1. **Kiểm thử là gì?**

**Kiểm thử phần mềm** là một cuộc kiểm tra được tiến hành để cung cấp cho các bên liên quan thông tin về chất lượng của [sản phẩm](https://vi.wikipedia.org/wiki/S%E1%BA%A3n_ph%E1%BA%A9m) hoặc [dịch vụ](https://vi.wikipedia.org/wiki/D%E1%BB%8Bch_v%E1%BB%A5) được kiểm thử. Kiểm thử có thể cung cấp cho doanh nghiệp một quan điểm, một cách nhìn độc lập về [phần mềm](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m) để từ đó cho phép đánh giá và thấu hiểu được những rủi ro trong quá trình [triển khai](https://vi.wikipedia.org/w/index.php?title=Tri%E1%BB%83n_khai&action=edit&redlink=1) phần mềm.

Trong kỹ thuật kiểm thử không chỉ giới hạn ở việc thực hiện một chương trình hoặc ứng dụng với mục đích đi tìm các [lỗi phần mềm](https://vi.wikipedia.org/w/index.php?title=L%E1%BB%97i_ph%E1%BA%A7n_m%E1%BB%81m&action=edit&redlink=1) (bao gồm các lỗi và các thiếu sót) mà còn là một quá trình phê chuẩn và xác minh một chương trình máy tính / ứng dụng / sản phẩm nhằm:

* Đáp ứng được mọi yêu cầu hướng dẫn khi thiết kế và phát triển phần mềm.
* Thực hiện công việc đúng như kỳ vọng.
* Có thể triển khai được với những đặc tính tương tự.
* Và đáp ứng được mọi nhu cầu của các bên liên quan.

Tùy thuộc vào từng phương pháp, việc kiểm thử có thể được thực hiện bất cứ lúc nào trong quá trình phát triển phần mềm. Theo truyền thống thì các nỗ lực kiểm thử được tiến hành sau khi các yêu cầu được xác định và việc lập trình được hoàn tất nhưng trong [Agile](https://vi.wikipedia.org/w/index.php?title=Agile&action=edit&redlink=1) (là một tập hợp các phương pháp phát triển phần mềm linh hoạt dựa trên việc lặp đi lặp lại và gia tăng giá trị) thì việc kiểm thử được tiến hành liên tục trong suốt quá trình xây dựng phần mềm. Như vậy, mỗi một phương pháp kiểm thử bị chi phối theo một quy trình phát triển phần mềm nhất định.

Kiểm thử không thể xác định hoàn toàn được tất cả các lỗi bên trong phần mềm. Thay vào đó, nó so sánh trạng thái và hành vi của sản phẩm với các oracle - các nguyên tắc hay cơ chế để phát hiện vấn đề. Các oracle này có thể bao gồm (nhưng không giới hạn ở) các đặc tả phần mềm, hợp đồng, sản phẩm tương đương, các phiên bản trước của cùng một sản phẩm, phù hợp với mục đích dự kiến nhằm đáp ứng sự kỳ vọng của người dùng, khách hàng, quy định của pháp luật hiện hành và các tiêu chuẩn liên quan khác.

Mục đích chính của kiểm thử là phát hiện ra các lỗi phần mềm để từ đó khắc phục và sửa chữa. Việc kiểm thử không thể khẳng định được rằng các chức năng của sản phẩm đúng trong mọi điều kiện, mà chỉ có thể khẳng định rằng nó không hoạt động đúng trong những điều kiện cụ thể. Phạm vi của kiểm thử phần mềm thường bao gồm việc kiểm tra mã, thực hiện các mã trong môi trường và điều kiện khác nhau, và việc kiểm thử các khía cạnh của mã: nó có làm đúng nhiệm vụ của nó hay không, và nó có làm những gì cần phải làm hay không. Trong môi trường phát triển phần mềm hiện nay, một đội kiểm thử có thể tách biệt với đội phát triển. Các thành viên trong đội kiểm thử giữ các vai trò khác nhau. Các thông tin thu được từ kiểm thử có thể được sử dụng để điều chỉnh quá trình phát triển phần mềm.

Mỗi sản phẩm phần mềm có một đối tượng phục vụ riêng. Ví dụ như đối tượng của phần mềm trò chơi điện tử là hoàn toàn khác với đối tượng của phần mềm ngân hàng. Vì vậy, khi một tổ chức phát triển hoặc đầu tư vào một sản phẩm phần mềm, họ có thể đánh giá liệu các sản phẩm phần mềm có được chấp nhận bởi người dùng cuối, đối tượng phục vụ, người mua, hay những người giữ vai trò quan trọng khác hay không. Và việc kiểm thử phần mềm là một quá trình nỗ lực để đưa ra những đánh giá này.

* 1. **Bối cảnh của các phần mềm.**

Ngày nay chúng ta đều đã biết đến các hệ thống phần mềm. Chúng ta bắt gặp chúng ở mọi nơi như trong nhà, nơi làm việc, các của hàng…chúng như là một phần cuộc sống của chúng ta. Chúng ta cũng dùng đến chúng cho rất nhiều công việc khác nhau và đi cùng với đó thì cũng đa cũng đã từng bắt gặp những lỗi không mong muốn của phần mềm như hệ thống phản hồi chậm, trang web không hiển thị đúng như yêu cầu, các chức năng không thự hiện đúng như mô tả, đó là một vài ví dụ cho những lỗi của một hệ thống phần mềm. Không phải mọi phần mềm đều chứa đựng những rủi do về lỗi như nhau và những lỗi xảy ra thì ảnh hưởng của chúng cũng không giống nhau nhưng khi chúng xảy ra thì đều là những thứ chúng ta không mong muốn. Khi chúng ta tìm hiểu về lỗi chúng ta phải xem xét khi nào lỗi đó có thể xảy ra và mức độ ảnh hưởng của lỗi khi chúng xảy ra.

Một vài lỗi chúng ta gặp khi sử dụng phần mềm thì có thể chấp nhận và không ảnh hưởng đến công việc. Nhưng một vài lỗi có thể có những ảnh hưởng lớn gây tổn hại nghiêm trọng nó làm chúng ta mất tiền bạc, thời gian, danh tiếc và có thể cả tính mạng. Ví dụ: Hệ thống hỗ trợ giao diện người dùng có các khuyết điểm về việc in ấn. Liệu nó có phải là một vấn để ? Nó có thể là một vấn đề bình thường nhưng nó cũng có thể trở thành một vấn đề hết sức nghiêm trọng, việc đó phụ thuộc vào website và các thiếu sót đó. Chả hạn website đó là website cá nhân của bạn thì nó hoàn toàn bình thường nhưng hãy tưởng tượng website đó là của một công ty nổi tiếng thì sao, khi khách hàng vào trang web đó thì họ sẽ cảm thấy không chuyên nghiệp và hiển nhiêu điều đó đã làm khác hàng một phần nào đó bớt uy tín hơn vào, và có khả năng công đó có thể mất những khách hàng tiềm năng.

* 1. **Nguyên nhân của các lỗi phần mềm.**

Tại sao đôi khi các phần mềm lại không hoạt động một cách chính xác? Chúng ta biết rằng con người mắc ra lỗi liệu điều đó có luôn đúng?

Nếu ai đó tạo ra một error hoặc là một mistake khi sử dụng phần mềm nó có thể trực tiếp dẫn đến các vấn đề, phần mềm không được sử dụng đúng cách và không hoạt động như chúng ta mong đợi. Con người xây dựng và thiết kế phần mềm có thể mắc những lỗi trong khi thiết kế và xây dựng chúng. Lỗi có nghĩa là có những thiếu xót trong bản thân phần mềm đó, chúng được gọi là Defect, Bugs hoặc là Faults.

Hãy suy nghĩ một chút về ảnh hưởng của các lỗi. Chúng ta đều đồng ý rằng mọi người đều có thể mắc lỗi. Các lỗi cũng có thể rất nghiêm trọng bởi hệ thống phần mềm thì thực sự phức tạp và mọi người gần như có thể mắc lỗi ở mọi công đoạn, một số lỗi được phát hiện và loại bỏ nhưng một số lỗi rất khó để tự mình tìm ra. Các khiếm khuyết như vậy trong phần mềm, hệ thống hoặc các tài liệu rất dễ dẫn đến thất bại của phần mềm khi xây dựng và gây thiệt hại khi được được ra sử dụng.

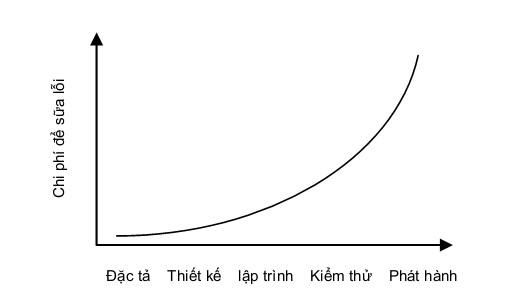
Chúng ta có thể mắc sai lầm khi chúng ta không có đủ kinh nghiệm, thông tin không chính xác, bị hiểu sai hoặc do chúng ta không cẩn thận, làm việc khi mệt mỏi, dưới áp lực của thời gian. Tất cả các nhân tố đó đều ảnh hưởng đến việc chúng ta đưa ra quyết định có chính xác không. Thêm vào đó, chúng ta nhiều khả năng mắc những sai xót khi chúng ta phải làm việc với những kĩ thuật khác nhau, phải giải quyêt các vấn đề tài chính, quy trình kinh doanh phúc tạp các thứ đó chi phối chúng ta làm chúng ta không tập chung và công việc chính và gây ra những lỗi. Lỗi cũng có thể bị gây ra bởi các điều kiện bên ngoài khác ví dụ như một chiếc radio bị phát nổ nó có thể có nhiều nguyên nhân bên ngoài như bị chập điện, do bị sét đánh, do bị đổ nước vào…Những lỗi này có thể phần nào đó được ngăn chặn hoặc hạn chế bớt bởi phương pháp kiểm thử.

Khí chúng ta nghĩ về những lỗi có thể xảy ra chúng ta phải xem xét các sai sót và các lỗi đó có thể xảy ra từ:

* Lỗi trong khi xây dựng, thiết kế và thực hiền phần mềm, hệ thống.
* Lỗi trong khi sử dụng hệ thống.
* Các điều kiện về môi trường bên ngoài.
* Các thiệt hại có chủ ý.
  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:



* 1. **Vai trò của kiểm thử trong phát triển, bảo trì và hoạt động của phần mềm.**

Chúng ta đã thấy rằng con người có thể gây ra những sai xót hay lỗi trong bất kì chu trình nào của một vòng đời sản phẩm và phụ thuộc vào những sai lầm mà kết quả có thể không có gì hoặc cực kì nghiêm trọng. Kiểm tra nghiêm ngặt là cần thiết trong quá trình phát triển và bảo trì để phát hiện, giảm các lỗi, nâng cao chất lượng hoạt động của hệ thống. Điều này bao gồm việc cả việc tìm kiếm những nơi trong giao diện người dùng mà người dùng có thể làm sai dữ liệu đầu vào hoặc là ảnh hưởng tớ dữ liệu đầu ra. Và những điểm yếu tiềm năng có thể bị tấn công. Thực hiện kiểm tra giúp chúng ta nâng cao chất lượng sản phẩm và dịch vụ nhưng nó cũng chỉ là một trong các phương pháp giúp xác minh và kiểm định được áp dụng cho các sản phẩm.

Chúng ta cũng có thể được yêu cầu thực hiện các kiểm thử để đáp ứng các yêu cầu của phần mềm hoặc về mặt pháp lý hoặc tiêu chuẩn công nghiệp cụ thể. Các tiêu chuẩn này có thể xác định những loại kĩ thuật chúng ta phải sử dụng hoặc tỉ lệ phần trăm mã chương trình cần phải được kiểm tra. Nó có thể làm bạn ngạc nhiên vì không phải lúc nào chúng ta cũng đi kiểm tra tất cả cá mã, vì nó có thể quá đắt đỏ so với những lối mà chúng ta tìm kiếm, do đó nhiều khi chúng ta chỉ đi tập chung vào những nơi hay xảy ra lỗi hoặc chúng ta cho rằng nó có khả năng mắc lỗi. Và mỗi nghành thì nó có những tiêu chuẩn kiểm thử riêng của mình, khi kiêm thử bạn bắt buộc phải tuân theo những yêu cầu tiêu chuẩn đó.

* 1. **Kiểm thử và chất lượng.**

Kiểm thử giúp chúng ta đo được chất lượng của phần mềm thông qua số lượng các lỗi được tìm thấy, số lượng test chạy, độ bao phủ của các tests đối với hệ thống. Chúng ta có thể thực hiện nó cho cả các thuộc tính chức năng lẫn các thuộc tính phi chức năng của phần mễm. Kiêm thử giúp tạo nên sự tin tưởng trong chất lượng phần mềm nếu nó tìm ra được ít lỗi điều đó khiến chúng ta hạnh phúc khi phần mềm của chúng ta đạt được các tiêu chuẩn cần thiết nhưng điều đó phải xảy ra với một quy trình kiêm thử tốt, tất nhiên với một quy trình kiêm thử kém nó sẽ không thể nào bao phủ hết tất cả các lỗi và để lại hậu quả sau này. Một chu trình kiểm thử được thiết kế tốt sẽ giúp chũng ta phát hiện được các nguy cơ và nếu tất cả các ca kiêm thử được thông qua thì nó giúp chúng ta tự tin hơn vào phần mềm và có thể khẳng định rằng mức độ rủi ro chung của việc sử dụng hệ thống này được giảm. Khi kiêm thử phát hiện ra các lỗi thì chất lượng của hệ thống phần mềm đấy sẽ được tăng lên sau khi chúng ta thực hiện các biện pháp sửa lỗi một cách đúng và chính xác.

**Chất lượng là gì?**

"Chất lượng" là một phạm trù phức tạp và có nhiều định nghĩa khác nhau. Có rất nhiều quan điểm khác nhau về chất lượng. Hiện nay có một số định nghĩa về chất lượng đã được các chuyên gia chất lượng đưa ra như sau:

" Chất lượng là sự phù hợp với nhu cầu" (theo Juran - một Giáo sư người Mỹ).

" Chất lượng là sự phù hợp với các yêu cầu hay đặc tính nhất định" Theo Giáo sư Crosby.

" Chất lượng là sự sự thoả mãn nhu cầu thị trường với chi phí thấp nhất" Theo Giáo sư người Nhật – Ishikawa.

Trong mỗi lĩnh vực khác nhau, với mục đích khác nhau nên có nhiều quan điểm về chất lượng khác nhau. Tuy nhiên, có một định nghĩa về chất lượng được thừa nhận ở phạm vi quốc tế, đó là định nghĩa của tổ chức tiêu chuẩn hóa quốc tiế. Theo điều 3.1.1 của tiêu chuẩn ISO 9000:2005 định nghĩa chất lượng là: "Mức độ đáp ứng các yêu cầu của một tập hợp có đặc tính vốn có"

Chất lượng là khái niệm đặc trưng cho khả năng thoả mãn nhu cầu của khách hàng. Vì vậy, sản phẩm hay dịch vụ nào không đáp ứng được nhu cầu của khách hàng thì bị coi là kém chất lượng cho dù trình độ công nghệ sản xuất ra có hiện đại đến đâu đi nữa. Đánh giá chất lượng cao hay thấp phải đứng trên quan điểm người tiêu dùng. Cùng một mục đích sử dụng như nhau, sản phẩm nào thoả mãn nhu cầu tiêu dùng cao hơn thì có chất lượng cao hơn.

Yêu cầu của khách hàng đối với sản phẩm hay dịch vụ thường là: tốt, đẹp, bền, sử dụng lâu dài, thuận lợi, giá cả phù hợp.

**Xem xét chất lượng phần mềm từ đầu đến cuối.**

Nếu phát sinh một vấn đề đối với việc Đảm bảo chất lượng phần mềm vào giai đoạn kiểm thử, khi mà tất cả các nhiệm vụ của nhóm phát triển đã hoàn tất, thì chi phí để nhóm khắc phục vấn đề là rất cao, đặt toàn bộ dự án vào nguy cơ rủi ro cao. Trong giai đoạn kiểm thử, các nhà phát triển làm hết sức mình để đảm bảo rằng mã của họ có ít sai sót. Sau đó, người kiểm thử làm việc chăm chỉ để phát hiện các lỗi có thể xảy ra trong phần mềm, trong khi các nhà quản lý và khách hàng hy vọng rằng phần mềm của họ đã sẵn sàng để phát hành ra thị trường.

Câu hỏi đặt ra là: Tại sao nhiều công ty phần mềm tập trung để thúc đẩy các nhóm phát triển đáp ứng thời hạn nghiêm ngặt và kết thúc với nhiều tính năng nhất có thể mà không quan tâm đến việc mã ứng dụng nghèo nàn như thế nào, bỏ qua số lượng lớn các sai sót bên trong các mã, gây ra các sai lầm trong kiến trúc, và bỏ qua các tài liệu?

Việc phát triển vội vã có thể tiết kiệm thời gian của nhóm nghiên cứu tại một thời điểm, nhưng cuối cùng nó khiến họ mất thêm thời gian để làm điều đó nếu có vấn đề phát triển mà không xem xét ngay từ đầu. Nó gây hậu quả lãng phí rất nhiều nguồn lực của nhóm để sửa chữa và tái cấu trúc mã của họ thay vì đầu tư nguồn lực vào việc hữu ích hơn. Nhóm phần mềm biết rõ điều đó, nhưng với những khách hàng khó tính và đội ngũ bán hàng khắt khe kèm theo cái tôi của một số nhà phát triển mà họ viết phần mềm không có bất kỳ sai sót nào, đội Đảm bảo chất lượng (QA) sẽ giúp kiểm tra lại mã nguồn khi hoàn thành.

* 1. **Kiểm thử bao nhiêu là đủ?**

Chúng ta đã biết rằng kiểm thử giúp chúng ta tìm ra các nguy cơ và giúp chúng ta nâng cao chất lượng phần mềm. Nhưng giờ có một câu hỏi đó là kiểm thử bao nhiêu là cần thiết? Chúng ta phải lựa chọn giữa việc kiêm thử tất cả mọi thứ, không kiêm thử hoặc chỉ một phần của phần mềm. Ngay lập tức các bạn sẽ trả lời rằng tất cả mọi thứ phải được kiêm tra. Chúng ta có muốn sử dụng một phần mềm mà không được kiểm tra tất cả không? Những thứ chúng ta cần xem xét liệu có phải kiêm tra hết?

Câu hỏi kiêm thử bao nhiêu là đủ thật sự phụ thuộc vào rất nhiều yếu tố khác nhau sau đây là một vài yếu tố như thế:

* Bao nhiêu kiểm thử là đủ nên dựa vào mức độ rủi ro, bao gồm rủi ro về kinh doanh, bảo mật, kỹ thuật và những hạn chế của dự án như là thời gian và ngân sách.
* Kiểm thử nên cung cấp đầy đủ thông tin cho các bên liên quan để quyết định việc phát hành phần mềm hoặc hệ thống sau khi đã được kiểm thử.
* Kiểm thử được chấm dứt khi một tiêu chí đầy đủ được đáp ứng.
* Dưới áp lực của công việc bao gồm thời gian, ngân sách đi cùng với đó là các giải pháp về công nghệ mà khách hàng yêu cầu. Cả khách hàng và nhà quản lý dự án chỉ muốn cung cấp một lượng kinh phí nhất định cho việc kiểm thử phần mềm. Do đó chúng ta không thể thực hiện toàn bộ được mà chỉ tập chung vào các chức năng chính.

1. **Các nguyên tắc kiểm thử.**

**Nguyên tắc 1:** Kiểm thử dưa 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 làm giảm xác suất lỗi chưa tìm thấy và vẫn còn trong phần mềm, thậm chí là không có lỗi nào, nó không phải là bằng chứng của sự chính xác.

**Nguyên tắc 2:** Kiểm thử đầy đủ là không thể

Kiểm thử mọi thứ là không thể thực hiện được. Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên mức độ ưu tiên chúng ta có thể tập chung việc kiểm thử vào một số điểm cần thiết.

**Nguyên tắc 3:** Kiểm thứ sớm

Để tìm được bug sớm, các hoạt động kiểm thử nên được bắt đầu càng sớm càng tốt trong quy trình phát triển phần mềm hoặc hệ thống, và nên tập trung vào các hoạt động đã định trước.

**Nguyên tắc 4:** Sự tập trung của lỗi

Nỗ lực kiểm thử nên tập trung một cách cân đối vào mật độ lỗi dự kiến và lỗi phá hiện ra sau đó trong các module. Một số ít các module thường chứa nhiều lỗi không phát hiện ra trong lúc kiểm thử trước khi phát hành, hoặc chịu trách nhiệm cho hầu hết các lỗi hoạt động của phần mềm.

**Nguyên tắc 5:** Nghich lý của thuốc trừ sâu

Nếu việc kiểm thử tương tự nhau được lặp lại nhiều lần thì cuối cùng sẽ có một số trường hợp kiểm thử sẽ không tìm thấy bất kì lỗi nào mới. Để khắc phục các trường hợp kiểm thử cần phải được xem xét và sửa đồi thường xuyên, và cần phải viết các test case mới và khác nhau để thực hiện nhiều phần khác nhau của phần mềm hoăc hệ thống để tìm ra lỗi tiềm ẩn nhiều hơn nữa.

Nguyên tắc này giống như việc trừ sâu trong nông nghiệp, nếu chúng ta cứ phun một loại thuốc với nồng độ giống nhau trong một khoảng thời gian dài thì có một số con sâu sẽ quen dần và cuối cùng việc phun thuốc giống như là tắm chúng vậy(nhờn thuốc) lúc đó chúng ta không thể diệt sạch chúng được. Do vậy để diệt sạch sâu một cách hiệu quả, người ta thường thay đổi các loại thuốc trừ sâu, mỗi loại chỉ dùng trong một khoảng thời gian ngắn.

**Nguyên tắc 6:** Kiểm thử theo cách ngữ cảnh độc lập

Nguyên tắc này là việc testing phụ thuộc vào ngữ cảnh, test trong nhiều ngữ cảnh khác nhau.

Để hiểu rõ hơn chúng ta xét ví dụ sau:

Vd: Một trương trình có tên “Calculator” có nhiều chức năng nhưng:

* Nếu test chương trình này cho mẫu giáo thì chỉ cần test cộng trừ là được.
* Nếu test chương trình này cho cấp 2 thì cộng trừ nhân chia.
* Nếu test chương trình này cho đại học thì tích phân, đạo hàm…

**Nguyên tắc 7:** Sự sai lầm về việc không có lỗi

Việc tìm và sửa chữa lỗi sẽ không giúp được gì nếu hệ thống được xây dựng xong nhưng không thẻ dùng được và không đáp ứng được nhu cầu của người dùng.

1. **Quy trình kiểm thử cơ bản.**
   1. **Kế hoạch kiểm thử và sử lý.**

Kế hoạch kiểm thử là hoạt động định nghĩa các đối tượng của kiểm thử và đặc tả các hoạt động kiêm thử với mục đích đạt được các đối tượng và nhiệm vụ.

Kiểm soát kiểm thử (Test control) liên quan tới các hoạt động cần thiết để đạt được các đối tượng và nhiệm vụ của dự án. Với mục đích kiểm soát được quá trình kiểm thử thì các hoạt động kiểm thử nên được điểu khiển trong suốt dự án. Kế hoạch kiêm thử mang lại những phản hồi từ các hoạt động điều khiển và kiểm soát.

* 1. **Phân tích và thiết kế kiểm thử.**

Đây là hoạt động xuyên suốt mà những đối tượng kiểm thử thông thường được chuyển đổi đến những điều kiện và các trường hợp kiểm thử rõ rệt.

Hoạt động phân tích và thiết kế kiểm thử có những nhiệm vụ sau:

* Review kiểm thử cơ bản (như các yêu cầu, các mức độ rủi ro, báo cáo phân tích rủi ro, kiến trúc, thiết kế, đặc tả giao diện)
* Đánh giá mức độ khả dụng của kiểm thử cơ bản và các đối tượng kiểm thử.
* Xác định thứ tự ưu tiên các điều kiện kiểm thử dựa trên phân tích các yếu tố kiểm thử, bản đặc tả cấu trúc của phần mềm.
* Thiết kế và đánh giá mức độ ưu tiên của các trường hợp kiểm thử.
* Xác định sự cần thiết của dữ liệu kiểm thử (test data) để hỗ trợ các điều kiện và các trường hợp kiểm thử.
* Thiêt kế môi trường kiểm thử và xác định bất cứ cơ sở hạ tầng và các test tool được yêu cầu.
  1. **Thực thi kiểm thử.**

Đây là hoạt động mà những kịch bản hoặc các thủ tục kiểm thử được đặc tả bởi dự kết hợp các trường hợp kiểm thử theo thứ tự riêng và bao gồm bất cứ thông tin cần thiết nào cho việc thực thi test, môi trường được thiết lập và việc kiểm thử được chạy.

Việc thực thi kiểm thử bao gồm những nhiệm vụ như dưới đây:

* Hoàn thiện, thực thi và phân thứ tự ưu tiên các trường hợp kiểm thử (bao gồm cả test data)
* Phát triển và phân thứ tự ưu tiên cho những thủ tục kiểm thử, tạo dữ liệu test, viết kịch bản test tự động…
* Tạo ra những bộ kiểm thử từ những test procedures để thực thi kiểm thử một cách hiệu quả
* Xác nhận môi trường kiểm thử đã được thiết lập chính xác
* Thực thi các test procedure cả cách thủ công hoặc sử dụng các tool kiêm thử tự động
* Đưa ra các kết quả mong muốn của quá trình thực thi test và các phiên bản của phần mềm
* So sánh kết quả thự tế và kết quả mong đợi
* Báo cáo sự khác biệt và phân tích chúng để tìm ra nguyên do lỗi (trong khi lập trình, trong dữ liệu đặc tả, trong tài liệu test hoặc lỗi khi kiểm thử được thực thi).
  1. **Đánh giá tiêu chí và thực hiện báo cáo.**

Đây là hoạt động mà việc thực thi kiểm thử được thẩm định lại các đối tượng đã được định nghĩa. Việc này nên được hoàn thành cho lỗi mức kiểm thử

Việc đánh giá này bao gồm các nhiệm vụ sau:

* Kiểm tra lại lịch sử test một lần nữa để tránh sót những Exit criteria đã được đặc tả trong kế hoạch test
* Thẩm định nếu cần nhiều kiểm thử hơn hoặc nếu exit criteria đó cần được thay đổi
* Viết báo cáo kết luận cho những bên liên quan tới dự án.
  1. **Những hoạt động chấm dứt quá trình kiểm thử.**

Kiểm thử chấm dứt khi hệ thống phần mềm được release, một dự án kiểm thử được hoàn thành (hoặc bị hủy) là một mốc milestone đạt được, hoặc sự bảo trì đã được hoàn thành.

1. **Tâm lý học của kiểm thử.**

Độ tin cậy của phần mềm là độ đo về mức độ tốt của các dịch vụ mà hệ cung cấp cho máy tính. Cần chú ý là người dùng không xét rằng các dịch vụ là quan trọng như nhau.

Có 4 thuộc tính quan trọng trong việc quyết định tính tin cậy của một phần mềm:

* Availability (Tính sẵn có): là thước đo về khả năng cung cấp các tính năng một cách liên tục cho người dùng đảm bảo không bị gián đoạn trong một khoảng xác định (có thể là thời gian hay số lượng yêu cầu)
* Reliability (Độ tin cậy): là thước đo về sự chính xác, hiệu quả trong việc cung cấp các tính năng người dùng mong muốn trong một khoảng thời gian sử dụng xác định.
* Safety (Sự an toàn): đảm bảo khi hoạt động hoặc khi bị sự cố phần mềm không gây ra thiệt hại cho người dùng.
* Security (Khả năng bảo mật): chống các can thiệp xấu từ bên ngoài nhằm phá hoại phần mềm hoặc đánh cắp thông tin.

Vai trò của 1 tester trong việc kiêm thử

* Đưa ra thông tin chính xác về lỗi và những thay đổi
* Hiểu quan niệm từ các nhà phát triển về lĩnh vực phần mềm
* Đưa code test đến một tiêu chuẩn thống nhất
* Tìm ra lỗi
* Yêu cầu kĩ thuật và kế hoạch test
* Thực hiện nghiêm túc báo cáo về lỗi
* Đưa ra các dự đoán về mức độ lỗi trong tương lai
* Cải thiện trình độ test
* Trách nhiệm của testers
* Theo dõi kế hoạch test, bàn thảo…giống như hồ sơ.
* Báo cáo lỗi khách quan và chính xác
* Kiểm tra lỗi có đúng không trước khi báo cáo
* Đánh giá khách quan rủi ro
* Thận trọng với những gì bạn báo cáo
* Báo cáo sự thật

1. **Mức độ ưu tiên của kiểm thử.**

Trong kiểm thử phần mềm thì hai khái niệm Độ ưu tiên (Priority) và Độ nghiêm trọng (Severity) cũng không quá xa lạ, đặc biệt là trong quản lý bug. Hai khái niệm trên đã trở nên quá quen thuộc và phổ biến đến nỗi chúng ta hầu như không phân biệt được ý nghĩa cũng như sự khác nhau giữa hai khái niệm đó. Mặc dù hai yếu tố này không phải là yếu tố sống còn trong quản lý bug nhưng việc hiểu đúng sẽ giúp chúng ta tiết kiệm thời gian cũng như làm công việc hiệu quả hơn.

* 1. **Các tiêu chí xếp hạng.**
* Test nơi lỗi là nghiêm trọng nhất
* Test nơi lỗi nhìn thấy rõ nhất
* Test nơi lỗi có khả năng nhất
* Hỏi khách hàng về yêu cầu
* Cái nào là quan trọng nhất cho doanh nghiệp của khách hàng
* Nơi thường xuyên thay đổi nhất
* Nơi đã xảy ra nhiều lỗi nhất
* Các nơi phức tạp, hoặc kĩ thuật quan trọng
  1. **Độ nghiêm trọng.**

Mức độ nghiêm trọng của một con bug thường chỉ mức độ tác động của con bug đó đến sản phẩm/ người dùng. Mỗi dự án hay sản phẩm có tiêu chí đánh giá độ nghiêm trọng khác nhau nhưng thông thường sẽ có 4-5 mức độ khác nhau từ nghiêm trọng nhất đến ít nghiêm trọng hơn:

**Mức độ 1:** Hệ thống sập, dữ liệu bị mất, ứng dụng không cài đặt được v.v.

**Mức độ 2:** Chức năng chính của sản phẩm không hoạt động

**Mức độ 3:** Chức năng phụ của sản phẩm không hoạt động

**Mức độ 4:** Bug nhỏ, không quan trọng

**Mức độ 5:** Yêu cầu cải tiến sản phẩm, thêm chức năng

Cũng nên lưu ý việc định nghĩa mức độ nghiêm trọng phụ thuộc vào sản phẩm khác nhau, mang tính tham khảo và tương đối.

Việc xác định được độ nghiêm trọng của con bug giúp nhà quản lí dự án, chủ sản phẩm có cái nhìn tốt hơn và thuận lợi hơn về tình hình chất lượng của sản phẩm. Số lượng bug là chưa đủ để đánh giá tình hình. Việc đội kiểm thử tìm được 50 con bug trong 1 tháng cũng không nói lên nhiều về tình hình chất lượng của sản phẩm. Tuy nhiên, nếu biết được trong 50 con bug đó có đến hơn 1 nửa là bug với độ nghiêm trọng ở cấp độ 1 và 2 sẽ hữu ích hơn nhiều. Ngoài ra, với góc độ của kỹ sư kiểm thử, độ nghiêm trọng cũng giúp “quảng cáo” cho con bug của mình từ đó sẽ gây được sự chú ý của mọi người và tăng cơ hội con bug đó được sửa.

Có một thực tế không lấy gì vui vẻ là việc xác định độ nghiêm trọng của con bug không hẳn lúc nào cũng mang tính chất trắng-đen và tuyệt đối. Sẽ không có gì ngạc nhiên nếu chúng ta cho rằng vấn đề này là nghiêm trọng trong khi chủ sản phẩm, nhà quản lý dự án lại không nghĩ như vậy. Không hẳn là họ đang cố tình “dìm hàng” chúng ta mà cũng có thể cách chúng ta cung cấp thông tin không thể hiện được đầy đủ mức độ nghiêm trọng của vấn đề. Hãy phân tích và cung cấp thêm thông tin để cho thấy tác động nghiêm trọng của con bug đối với sản phẩm cũng như người dùng cuối như thế nào như xảy ra trên nhiều môi trường; lặp đi lặp lại; có khả năng ảnh hưởng đến các thành phần, chức năng khác; hình ảnh thương mại của công ty v.v. Và dĩ nhiên, nếu vấn đề không thực sự nghiêm trọng, chúng ta cũng không có lí do gì để làm cho lớn chuyện. Việc hiểu rõ sản phẩm, người dùng cùng khả năng phân tích, suy luận sẽ giúp chúng ta làm tốt khâu này.

* 1. **Độ ưu tiên.**

Như chúng ta đã biết nếu đã là bug thì sẽ phải sửa. Tuy nhiên, có một thực tế là đội phát triển khó có thể sửa hết tất cả bug một lượt cũng như không đáng để sửa hết tất cả các bug. Do đó, đội phát triển sẽ phải cần đến độ ưu tiên của con bug để biết được bug nào cần được sửa trước bug nào sau. Độ ưu tiên của con bug cũng thường được chia thành 3-4 cấp độ:

**Mức độ 1:** Cao – Bug sẽ phải sửa ngay lập tức

**Mức độ 2:** Trung bình – Bug sẽ sửa trong bản cập nhật lần tới

**Mức độ 3:** Thấp – Bug không cần sửa trong bản cập nhật lần tới, có thể sửa nếu có thời gian

Tương tự mức độ nghiêm trọng, mức độ ưu tiên cũng như ý nghĩa của chúng có thể sẽ khác nhau ở những sản phẩm, dự án khác nhau.

Thế chúng ta sẽ dựa vào đâu để xác định độ ưu tiên? Bug nào sửa trước bug nào sửa sau (hoặc không sửa)? Quá dễ, dựa vào độ nghiêm trọng của bug. Bug nào nghiêm trọng nhất, tác động đến người dùng nhiều nhất thì sẽ được ưu tiên sửa trước. Bug nào ít nghiêm trọng hơn sẽ được sửa sau. Đúng…nhưng không phải lúc nào cũng đúng. Giả sử chúng ta tìm được một bug làm sập hệ thống. Quá tuyệt đúng không nào. Chúng ta đánh giá mức độ nghiêm trọng ở Mức độ 1 (rất là nghiêm trọng) và độ ưu tiên 1 (cần được sửa gấp). Nhưng hôm sau, độ ưu tiên của bug đó được điều chỉnh xuống thấp trong khi con bug bắt lỗi chính tả của thằng bạn lại được đánh giá có độ ưu tiên cao để sửa. Chúng ta buồn rầu, thất vọng, khó chịu và quyết phải hỏi cho ra lẽ và được giải thích rằng “Mặc dù bug đó làm sập hệ thống nhưng khả năng người dùng bị lỗi đó là rất thấp do để làm ra bug đó bạn phải trải qua vài chục bước hay bug đó chỉ xảy ra trên một môi trường cụ thể cũng như rất ít người dùng chạy sản phẩm trên môi trường đó”. Suy cho cùng đó là một quyết định liên quan đến kinh doanh và hầu như chúng ta không thể thay đổi được quyết định đó. Có một thực tế phải thừa nhận là kỹ sư kiểm thử chúng ta biết rất ít hay thậm chí không thể biết được khối lượng công việc của đội phát triển, chi phí của dự án cũng như những quyết định kinh doanh mang tính chiến lược của nhà đầu tư, quản lý dự án hay chủ sản phẩm. Điều đó cũng không có gì là quá tệ đối với một kỹ sư kiểm thử. Nó chỉ liên quan đến vai trò và trách nhiệm công việc của kỹ sư kiểm thử. Xác định độ ưu tiên được khuyến khích đối với kỹ sư kiểm thử nhưng không phải bắt buộc. Đó là lí do tại sao ở một số dự án thậm chí kỹ sư kiểm thử được yêu cầu không xác định độ ưu tiên cho con bug và độ ưu tiên chỉ được xác định sau buổi họp đánh giá bug. Điều này cũng không có gì là vô lí.