

**ĐẠI HỌC CÔNG NGHỆ THÔNG TIN KHOA CÔNG NGHỆ PHẦN MỀM**

**KIỂM THỬ PHẦN MỀM**

(Software Testing)

**GVHD: Dương Minh Đức**

**15520059 – Đoàn Văn Châu**

**15520461 – Phan Văn Lượm**



**Tài liệu tham khảo**

[1]. Roger S. Pressman. Software Engineering, 5th edition. USA, McGraw-Hill, 2003.

Tài liệu tham khảo chính:

[2]. Cem Kaner, Jack Falk, Hung Q. Nguyen. Testing Computer Software, 2nd edition. Canada, Wiley, 1999.

[3]. Hung Q. Nguyen. Testing applications on the web. USA, Wiley.

[4] Paul Ammann, Jeff Offutt (2008): Introduction to Software Testing, Cambridge University Press.

]5] Glenford J. Myers (2004): The art of Software Testing,

John Wiley & Son.

[6] LogiGear (2009): Basic Software Testing Skills, LogiGear Corporation.



**Nội dung môn học**

## Bài 1: Tổng quan kiểm thử phần mềm

* Bài 2: Quy trình kiểm thử phần mềm
* Bài 3: Các cấp độ kiểm thử
* Bài 4: Các loại kiểm thử
* Bài 5: Các kỹ thuật kiểm thử
* Bài 6: Kiểm thử tự động
* Bài 7: Quản lý chất lượng phần mềm



**HÌNH THỨC KIỂM TRA**

* + **Hình thức kiểm tra:** (tỷ lệ 100%)

##### LÝ THUYẾT: 50% (trắc nghiệm + tự luận)

* + - QUÁ TRÌNH : làm bài thu hoạch làm việc nhóm 50%
    - Chuyên cần: 10% chuyên cần và tích cực thảo luận diễn đàn và làm bài tập cá nhân.

**Bài 1**

**TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM**

##### 1.1 Phần mềm và chất lượng phần mềm, SQA



**BÀI 1: Tổng quan kiểm thử phần mềm**

* 1.2 Các yếu tố ảnh hưởng đến chất lượng phần mềm
* 1.3 Khái niệm kiểm thử phần mềm
* 1.4 Mục tiêu kiểm thử
* 1.5 Tầm quan trọng của kiểm thử
* 1.6 Các nguyên tắc trong kiểm thử
* 1.7 Một số khái niệm liên quan
* 1.8 Các đối tượng thực hiện kiểm thử
* 1.9 Các điểm cần lưu ý khi kiểm thử
* 1.10 Các hạn chế của kiểm thử 6



**1.1 Phần mềm và chất lượng phần mềm**

* + Phần mềm và các đặc trưng
  + Các khái niệm vễ lỗi, sai sót, hỏng hóc
  + Nguyên nhân gây ra lỗi phần mềm
  + Chất lượng phần mềm
  + Đảm bảo chất lượng phần mềm



* + **Theo định nghĩa của IEEE: B**ao gồm các chương trình máy tính, các thủ tục, các tài liệu có thể liên quan và các dữ liệu liên quan đến hoạt động của hệ thống máy tính
  + **Theo định nghĩa của ISO:** 4 thành phần cơ bản của phần mềm:

##### Chương trình máy tính (code)

* + - Các thủ tục
    - Tài liệu
    - Dữ liệu cần thiết để vận hành phần mềm



* + **Đặc trưng của phần mềm:**
    - Phần mềm được thiết kế, chế tạo như các loại sản phẩm công nghiệp khác, nhưng **không được định hình trước**

##### Quá trình phát triển phần mềm quyết định giá thành và chất lượng của nó

* + - Các phần mềm chỉ thực sự được tìm ra lỗi trong pha phát triển.



* **Đặc trưng của phần mềm:**
* *Có tính phức tạp cao và luôn thay đổi.*
* Phần mềm là một hệ thống logic với nhiều khái niệm và các mối liên hệ logic khác nhau => mỗi một vòng lặp với một giá trị khác nhau là cơ hội để tìm ra lỗi của phần mềm
* Thay đổi theo nhu cầu của người dùng
* Thay đổi để đáp ứng môi trường vận hành
* Phần mềm không nhìn thấy được
* Phần mềm không nhìn thấy được mà chỉ có thể nhận biết qua sự mô tả từ những khía cạnh khác nhau (sơ đồ điều khiển, mô hình luồng dữ liệu, mô hình tương tác…)
* Do đặc trưng này nên khả năng tìm ra lỗi một cách nhanh chóng là không thể

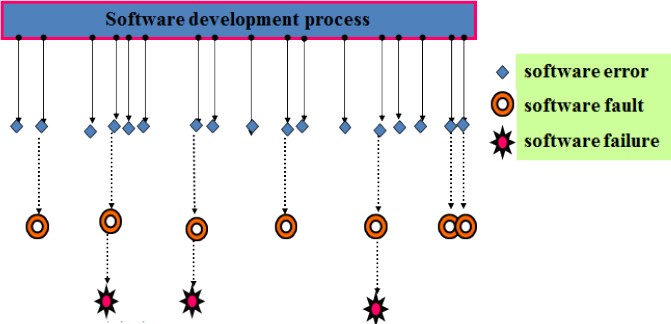


* **Lỗi phần mềm (software error)**
* Là lỗi do con người gây ra (thường là các lập tr.nh viên)
* Lỗi phần mềm có thể là lỗi cú pháp hoặc lỗi logic
* **Sai sót của phần mềm (software fault)**
  + Sai sót của phần mềm không phải lúc nào cung do lỗi phần mềm
  + Có thể có sai sót do dư thừa hoặc bỏ sót yêu cầu phần mềm (từ khâu khảo sát, phân tích, đưa ra yêu cầu phần mềm bị thừa hoặc bị sót so với yêu cầu của khách hàng)
* **Hỏng hóc của phần mềm(software failure)**
  + Một sai sót của phần mềm dẫn đến hỏng hóc khi nó sai sót đó

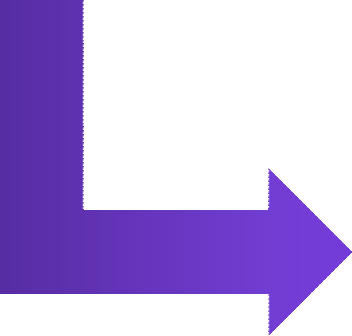
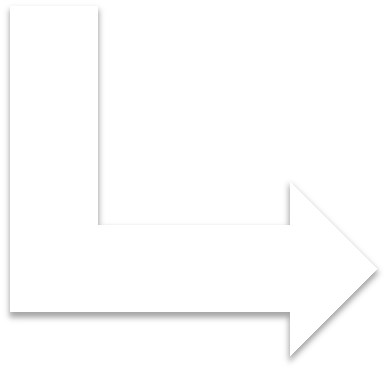
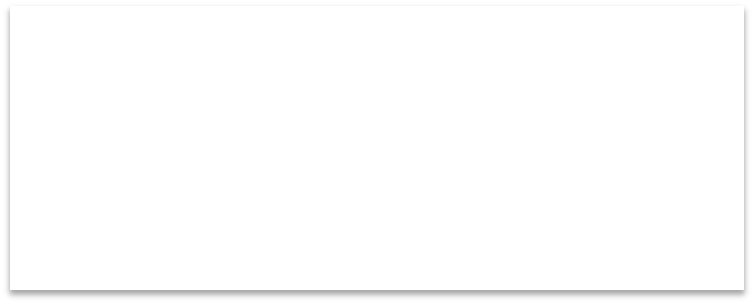
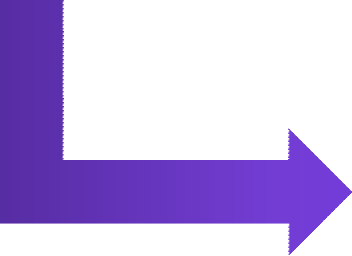
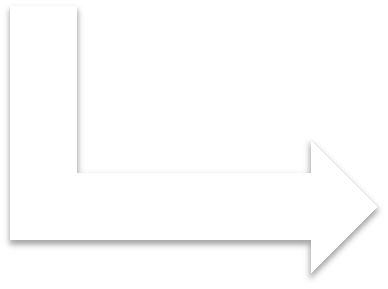
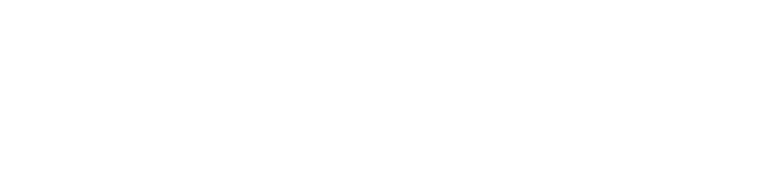
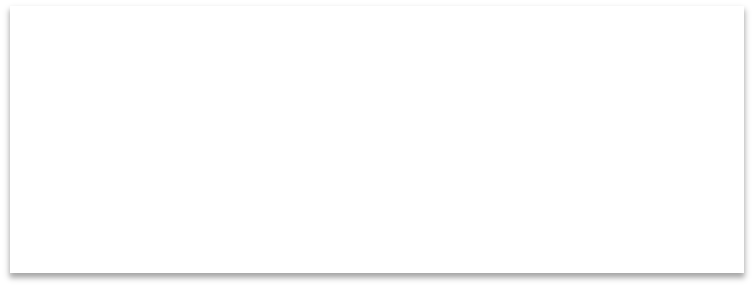
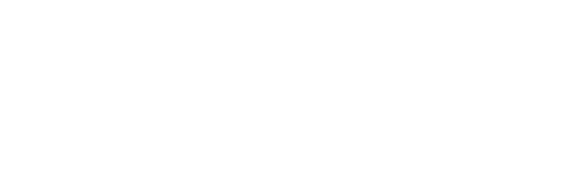
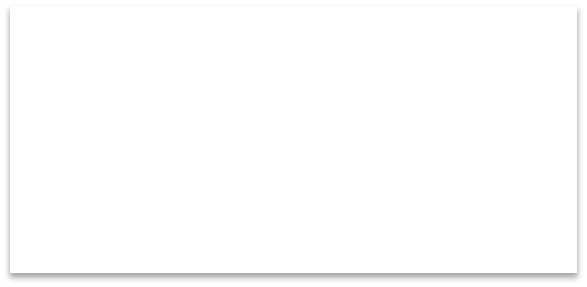
bị phát hiện

* + Một sai sót của phần mềm nếu không bị phát hiện hoặc ko gây ảnh hưởng tới phần mềm thì sẽ không được coi là hỏng hóc11 của pm





12



A developer makes an ERROR

… and injects a FAULT into the

software

… and the fault causes software to FAIL



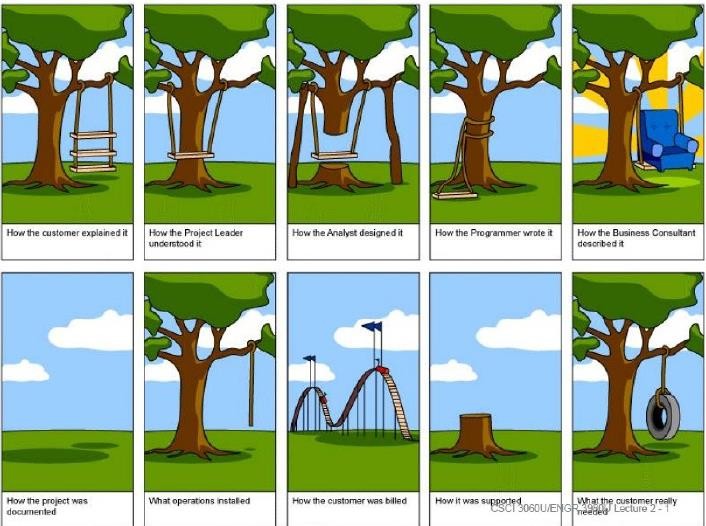
**1.1.3 Các nguyên nhân gây ra lỗi phần mềm**

* 1. **Định nghĩa sai yêu cầu của khách hàng**

##### Đây được coi là gốc rễ của việc gây ra lỗi phần mềm

* + Hiểu sai yêu cầu của khách hàng
  + Yêu cầu của khách hàng không được làm rõ
  + Triển khai phần mềm thiếu yêu cầu của khách hàng
  + Khách hàng đưa ra quá nhiều yêu cầu không cần thiết và không liên quan







* 2. Thất bại trong việc giao tiếp giữa người phát triển và khách hàng
  + Có sự không hiểu cấu trúc của tài liệu yêu cầu phần

mềm

* + Không nắm bắt được những thay đổi được viết trong tài liệu yêu cầu
  + Những thay đổi được yêu cầu từ khách hàng nhưng ko được lưu dưới dạng văn bản
* - Thiếu sự chú ý tới:
  + Thông điệp của khách hàng đề cập tới việc thay đổi

yêu cầu

* + Trả lời của khách hàng tới những câu hỏi mà developer đặt ra



### 3. Tạo ra độ lệch cố ý trong yêu cầu phần mềm

##### Lập trình viên sử dụng những module phần mềm có sẵn từ những dự án trước mà không thay đổi cho phù hợp với yêu cầu của dự án mới nhằm tiết kiệm thời gian

* + Bỏ qua một vài yêu cầu của phần mềm do thời gian quá gấp hoặc chi phí không đủ đáp ứng.



* 5. Lỗi mã hóa
  + Lỗi logic
  + Lỗi cú pháp
  + Lỗi thời gian chạy
* 6. Không tuân theo các tài liệu và cấu trúc code
  + Không tuân theo các chuẩn tài liệu (templates…)
  + Không tuân theo các cấu trúc mã hóa
* 7. Rút ngắn quá trình kiểm thử phần mềm
  + Do áp lực về thời gian, tiến độ hoàn thành dự án
  + Lập kế hoạch kiểm thử không đầy đủ
  + Không báo cáo đầy đủ các lỗi
  + Báo cáo không chính xác lỗi



* 8. Lỗi thủ tục

Chỉ dẫn cho người dùng những hoạt động cần thiết ở một quá trình. Nó quan trọng trong các hệ thống pm phức tạp khi quá trình xử lý được thực hiện qua nhiều bước. Mỗi bước có nhiều dạng dữ liệu và cho phép kiểm tra kết quả trung gian

* 9. Lỗi tài liệu
  + Sai sót trong hồ sơ thiết kế
  + Sai sót trong việc lập tài liệu hướng dẫn sử dụng
  + Các danh sách chức năng không có trong phần mềm

nhưng lại có trong tài liệu



* **Theo quan điểm của người dùng**: sản phẩm phù hợp với mục đích sử dụng của người dùng

### Theo quan điểm của nhà cung cấp sản

#### **phẩm**: sản phẩm đạt được các tiêu chí đánh giá do nhà cung cấp đề ra

* **Theo quan điểm của nhà sản xuất phần mềm**: sản phẩm đáp ứng đầy đủ các tiêu chí đề ra trong bản đặc tả.



* **Định nghĩa của IEEE**:
* Chất lượng phần mềm là:
  + Mức độ mà một hệ thống, thành phần hoặc quá trình đáp ứng yêu cầu quy định
  + Mức độ và một hệ thống, thành phần hoặc quá trính đáp ứng nhu cầu của người sử dụng hoặc mong đợi của khách hàng.
* **Theo cách tiếp cận của ISO:**
* Chất lượng toàn diện của phần mềm cần phải được quan tâm từ:
  + Chất lượng quy trình
  + Chất lượng phần mềm nội bộ (chất lượng trong)
  + Chất lượng phần mềm đối chiếu với yêu cầu người dùng (chất lượng ngoài)
  + Chất lượng phần mềm trong sử dụng (chất lượng sử dụng2)1
* Đảm bảo chất lượng phần mềm:



**1.1.5 Đảm bảo chất lượng phần mềm**

* Thiết lập một **tập hợp** các họat động có chủ đích và có hệ thống nhằm mang lại sự tin tưởng sẽ đạt được chất lượng đúng theo yêu cầu.
  + Đảm bảo dự án phần mềm sẽ hoàn thành đúng đặc tả, theo chuẩn mực định trước và các chức năng đòi hỏi, không có hỏng hóc và các vấn đề tiềm ẩn.
  + Điều khiển và cải tiến tiến trình phát triển phần mềm ngay từ khi dự án bắt đầu. Nó có tác dụng “phòng ngừa” cái xấu, cái kém chất lượng.
  + Mục tiêu: thỏa mãn khách hàng (Thời gian+Ngân sách+Chất lượng)

22

# Tester & QA



* KS kiểm định (Tester) có nhiệm vụ khảo sát, chạy thử để bảo đảm PM thỏa mãn các yêu cầu về chức năng và khả năng vận hành mà nó phải có, báo cáo các lỗi nếu có để các bộ phận liên quan chỉnh sửa. Công việc của KS kiểm định liên quan đến sản phẩm (product).
* KS chất lượng (QA) có nhiệm vụ giám sát để bảo đảm các tiêu chuẩn và quy trình sản xuất PM được định nghĩa và tuân thủ nghiêm túc, hướng đến mục tiêu các sản phẩm (SP) trung gian cũng như SP sau cùng của dự án thỏa mãn các tiêu chuẩn và yêu cầu đã định trước đó. Công việc của KS chất lượng liên quan đến quy trình (process).
* Ví dụ: Kiểm tra để bảo đảm các giải thuật khi viết code phải được chú thích rõ ràng, các Yêu cầu khách hàng được xem xét cẩn thận và mọi người hiểu giống nhau, các tài liệu đi kèm SP được kiểm tra trước khi gửi cho khách hàng 23



**1.2 Các yếu tố ảnh hưởng đến chất lượng**

#### Có ba yếu tố ảnh hưởng tới chất lượng phần mềm (tam giác chất lượng)

##### Con người

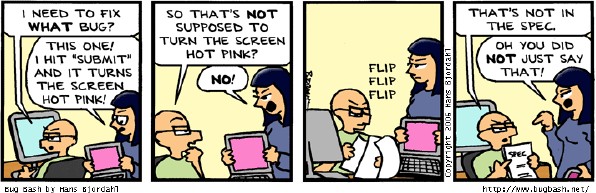
* + Quy trình
  + Công cụ



**1.2 Tiếp**

* Khoảng cách giữa yêu cầu người dùng và bản đặc tả yêu cầu hệ thống:
  + Không hiểu rõ yêu cầu của người dùng
  + Bỏ qua yêu cầu
  + Thiếu yêu cầu
  + Không đồng bộ về các phiên bản của tài liệu yêu cầu người dùng và tài liệu đặc tả
  + Bản đặc tả có thêm những yêu cầu không xuất phát từ người dùng





* Khoảng cách giữa bản đặc tả và sản phẩm:



* + Hiểu sai yêu cầu đặc tả do trong bản đặc tả có những chỗ diễn đạt chưa rõ ràng cụ thể.
  + Có các yêu cầu được đưa thêm vào trong quá trình phát triển nhưng không được thêm vào bản đặc tả.
  + Có sự thay đổi yêu cầu trong quá trình phát triển nhưng không được cập nhật vào bản đặc tả
  + Các tính năng mới được thêm vào bởi mục đích riêng của người phát triển
  + Các yêu cầu có trong bản đặc tả nhưng bị bỏ qua do quá khó để thực hiện



**1.2 Tiếp**

* Khoảng cách giữa yêu cầu người dùng và sản phẩm:
  + Khoảng cách này xuất hiện do sản phẩm làm ra không thỏa mãn yêu cầu người dùng
  + Độ lệch này phụ thuộc vào hai cạnh còn lại của tam giác chất lượng
  + Đây là độ lệch gây tốn kém nhất để sửa chữa
* Theo Glenford Myers:



**1.3 Khái niệm kiểm thử**

* + Kiểm thử là quá trình vận hành chương trình để tìm ra lỗi
* Theo IEEE: 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 và đưa ra 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 29

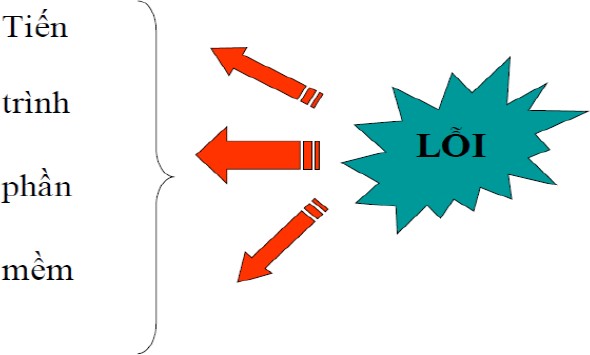


**1.4 Mục tiêu của kiểm thử**

* Tìm ra được càng nhiều lỗi càng tốt trong điều kiện về thời gian đã định và nguồn lực sẵn có
* Chứng minh rằng sản phẩm phần mềm phù hợp với các đặc tả của nó.
* Xác thực chất lượng kiểm thử phần mềm đã dùng chi phí và nỗ lực tối thiểu
* Thiết kế tài liệu kiểm thử một cách có hệ thống và thực hiện nó sao cho có hiệu quả, tiết kiệm được thời gian công sức.

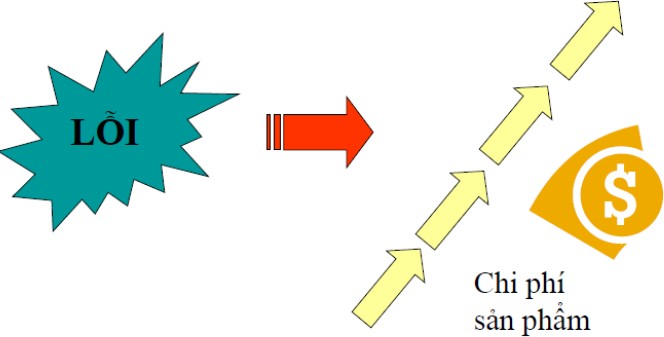


**1.5 Tầm quan trọng của kiểm thử**



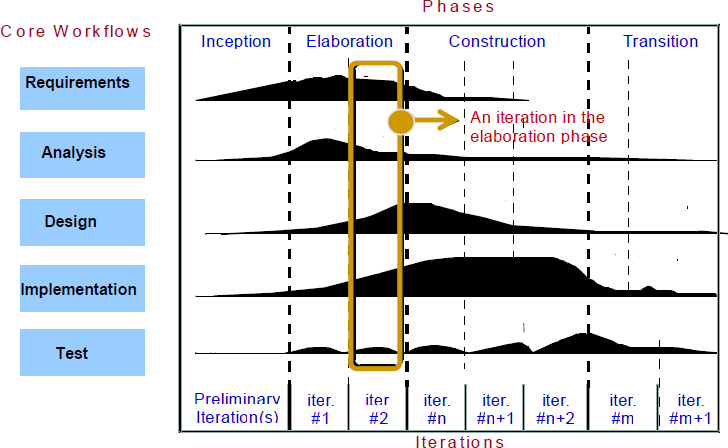


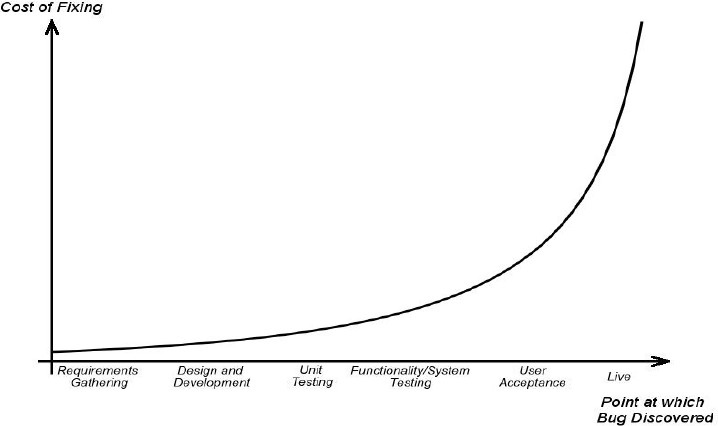
**1.5 Tầm quan trọng của kiểm thử**





**Qui trình phát triển phần mềm RUP**





% Defects Introduced in this phase

$14,000

85%

$1000

$25

$130

$250

Percentage of Bugs

% Defects found in

in this phase

$ Cost to repair defect in this phase

Coding Unit Test

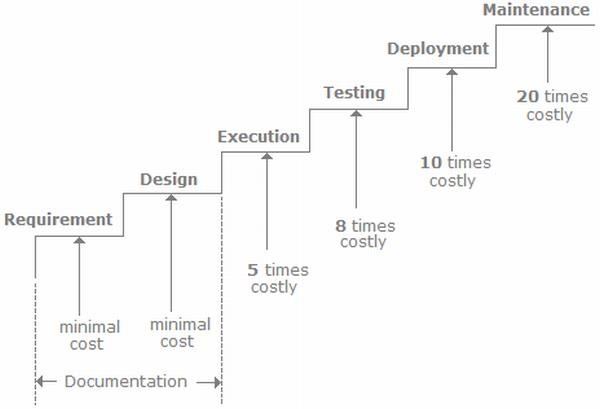
Funct Test

Field Test

Post Release

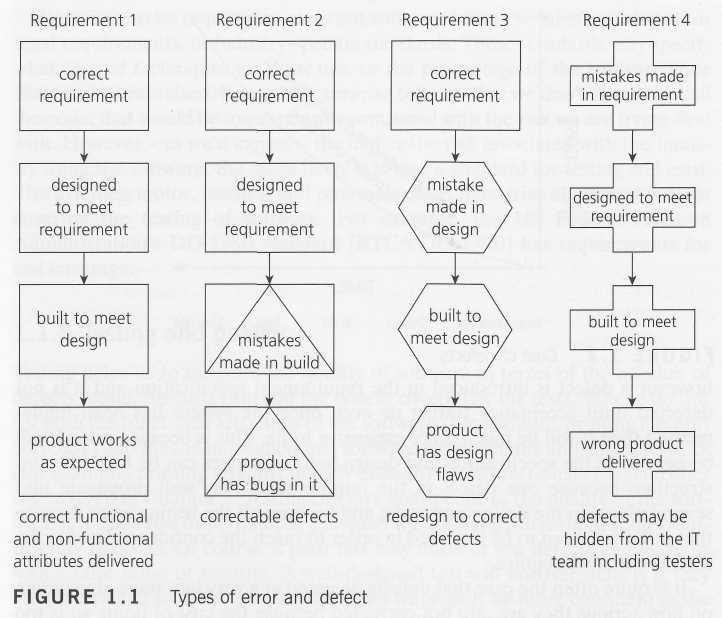
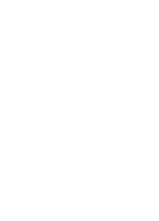
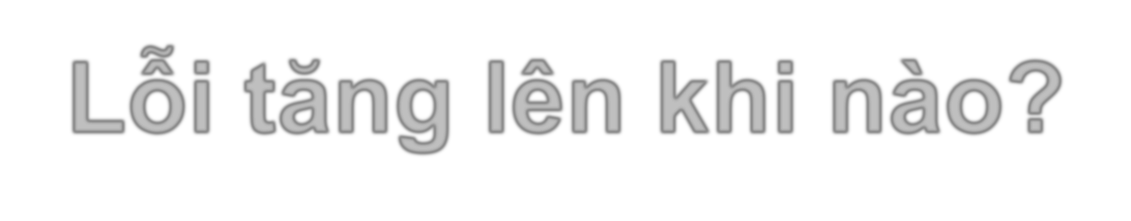
*Source: Applied Software Measurement, Capers Jones, 1996*





36

# Lỗi tăng lên khi nào?



37



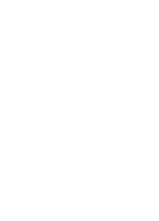
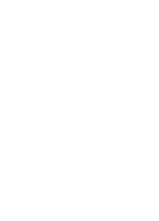
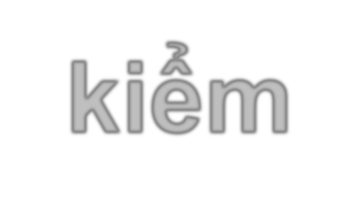
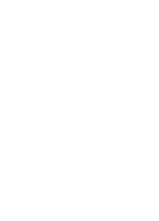
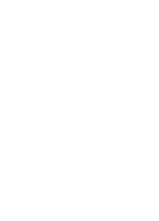
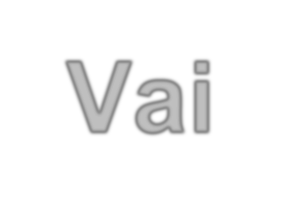
###### Những người phát triển phần mềm cho rằng:

* + Kiểm thử chỉ 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 hiện để chứng tỏ chương trình đã làm được các chức năng cần có.
* Những ý kiến trên về kiểm thử đã đầy đủ?
  + Kiểm thử còn để tìm ra lỗi và sửa chữa các lỗi đó nhằm tăng độ tin cậy cho phần mềm.

38



* Tại sao cần thực hiện kiểm thử?
* Để xem xét chất lượng sản phẩm
* Để phát hiện ra lỗi
* Ví dụ: Khách hàng có thể rút tiền ở máy ATM với số tiền tối đa là 250$/1 giao dịch
  + Người kiểm thử 1:
  + Thử 3 lần với 3 yêu cầu: 50$, 150$, 250$ thấy máy đều nhả ra số tiền chính xác, kết luận chức năng rút tiền hoạt động đúng yêu cầu của khách hàng là yêu cầu rút ra bao nhiêu đều trả về đúng bây nhiêu tiền.
  + Người kiểm thử 2:
  + Yêu cầu số tiền là 300$, máy vẫn nhả ra đúng 300$ mà ko đưa ra thông báo số tiền rút bị quá hạn, như vậy là có lỗi mà người kiểm thử 1 ko tìm ra được. 39



**Vai trò kiểm thử**

* Vai trò kiểm thử trong suốt quy 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.
* Trong kiểm thử có 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 điều không thể
3. Nên thực hiện kiểm thử càng sớm càng tốt
4. Sự phân cụm của các lỗi
5. Nghịch lý thuốc trừ sâu
6. Kiểm thử theo các ngữ cảnh độc lập
7. Sự sai lầm về việc không có lỗi

**1.7 Phân loại kiểm thử**

* Phân loại kiểm thử dựa trên các yếu tố:
  + Mục đích kiểm thử
  + Chiến lược kiểm thử
  + Phương pháp kiểm thử
  + Kỹ thuật kiểm thử
* Kiểm thử đơn vị, module



**1.7.1 Dựa vào mục đích kiểm thử**

* Kiểm thử cấu hình
* Kiểm thử sơ lược (smoke testing)
* Kiểm thử chức năng
* Kiểm thử tích hợp
* Kiểm thử hồi quy
* Kiểm thử hệ thống
* Kiểm thử tải dữ liệu (load testing)
* Kiểm thử tải trọng (stress testing)
* Kiểm thử hiệu suất (performance testing)
* Kiểm thử chấp nhận (UAT)
* Kiểm thử bảo mật (security testing) 43



**1.7.2 Dựa vào chiến lược kiểm thử**

* **Kiểm thử thủ công:**
  + Thực hiện kiểm thử mọi thứ bằng tay, từ viết test case đến thực hiện test.
* **Kiểm thử tự động**:
  + Thực hiện một cách tự động các bước trong kịch bản kiểm thử bằng cách dùng một công cụ trợ giúp
  + Kiểm thử tự động nhằm tiết kiệm thời gian kiểm thử
* Kiểm thử tĩnh:



**1.7.3 Dựa vào pp tiến hành kiểm thử**

* + Một hình thức của kiểm thử mà phần mềm không được sử dụng thực sự.
  + Thường không kiểm thử chi tiết mà chủ yếu kiểm tra tính đúng đắn của code, thuật toán hoặc tài liệu
  + Các hoạt động: Đi xuyên suốt (walk through), thanh tra (inspection)
* Kiểm thử động:
  + Một hình thức kiểm thử phần mềm chạy mã lập trình thực tế trong các tình huống, diễn ra khi bản thân chương trình đó đang được sử dụng
  + Kiểm thử động có thể bắt đầu trước khi chương trình đã hoàn tất.



**1.7.4 Dựa vào kỹ thuật kiểm thử**

* Kiểm thử hộp trắng
  + Kiểm thử theo góc nhìn thực hiện
  + Cần có kiến thức về chi tiết thiết kế và thực hiện bên

trong

* + Kiểm thử dựa vào phủ các lệnh, các nhánh, phủ các điều kiện con
* Kiểm thử hộp đen
  + Kiểm thử theo góc nhìn sử dụng
  + Kiểm thử dựa trên các yêu cầu và đặc tả sử dụng thành phần phần mềm
  + Không đòi hỏi kiến thức về chi tiết thiết kế và thực hiện ở bên trong chương trình



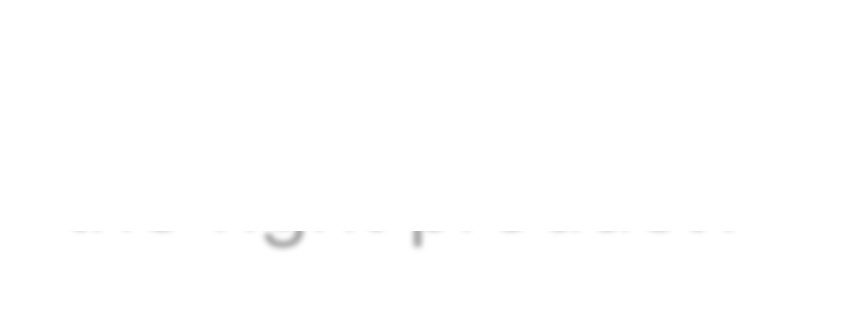
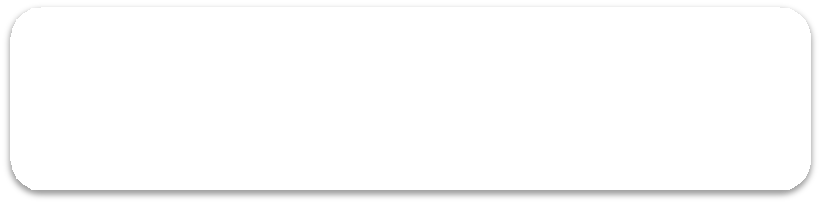
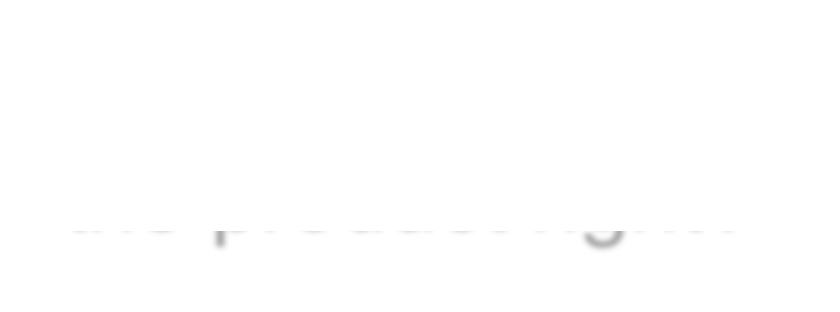
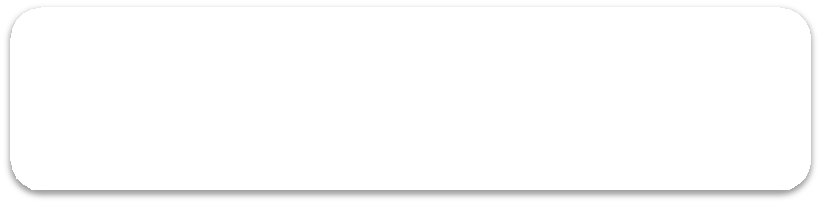
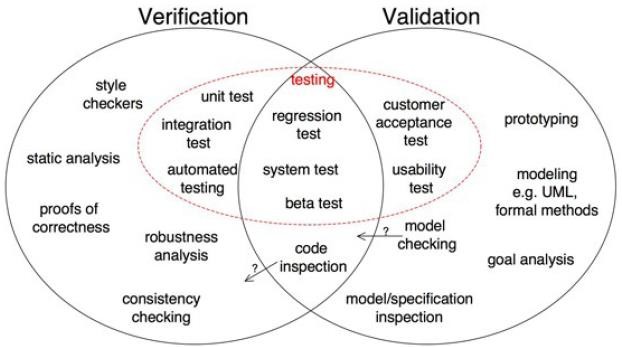
* Xác minh (Verification)
  + **Xác minh** là quy trình xác định xem sản phẩm của một công đoạn trong quy trình phát triển phần mềm có thỏa mãn các yêu cầu đặt ra trong công đoạn trước hay không?(Ta có đang xây dựng đúng sản phẩm mà được đăc tả không?)
  + Xác minh quan tâm tới việc ngăn chặn lỗi giữa các công đoạn
  + Xác minh thường là hoạt động kỹ thuật và nó có sử dụng các kiến thức về các yêu cầu, các đặc tả rời rạc của phần mềm
  + Các hoạt động của xác minh bao gồm: Kiểm thử (Testing) và Rà soát loại (Review)
* **Thẩm định (Validation)**

##### Là tiến trình nhằm chỉ ra toàn bộ hệ thống đã phát triển xong phù hợp với tài liệu mô tả yêu cầu. Thẩm định là quá trình kiểm chứng chúng ta xây dựng phầm mềm có đúng theo yêu cầu khách hàng không?

* + Thẩm định chỉ quan tâm đến sản phẩm cuối cùng không còn lỗi



* Xác định và thẩm định (vertification & Validation)



A we producing the product right?

A we producing the right product?



50

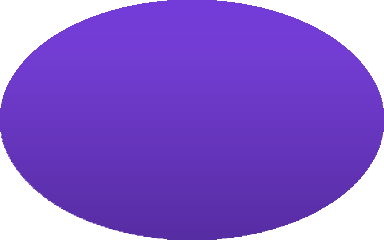
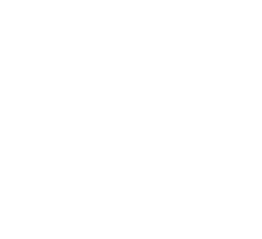
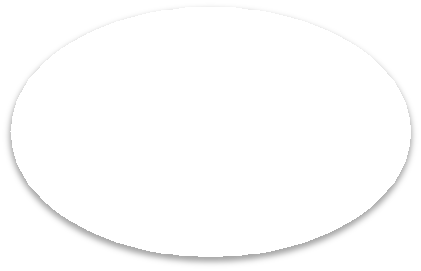
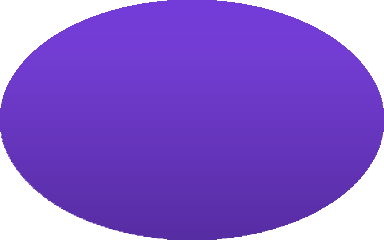
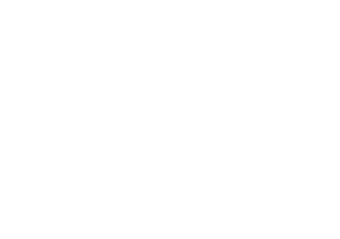
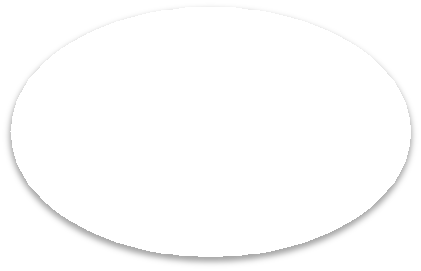
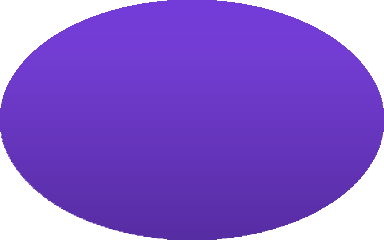
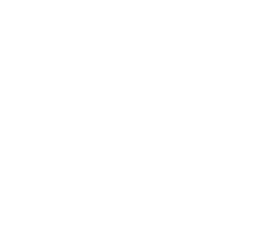
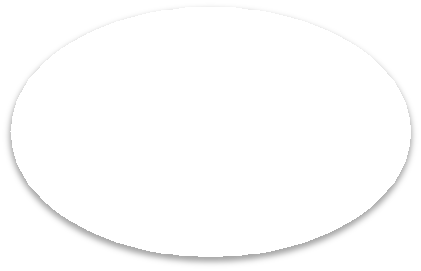
**Beta testing**

**Database testing**

**Re-testing**

**Mobile testing**

Test Level



Test

Test

**Alpha testing**

**Regression testing**

**Stress testing**

**SQL testing**

Method

Type

**GUI testing**

**Web testing**

**… testing**

**Smoke testing Installation testing**

**Sanity testing Load testing**



* **Dữ liệu kiểm thử** (test data): Dữ liệu cần cung cấp để phần mềm có thể thực thi để kiểm thử
* **Kịch bản kiểm thử** (test scenario): Các bước thực hiện khi kiểm thử
* **Kỹ sư kiểm thử** (tester): người thực hiện kiểm thử



###### **Ca kiểm thử** (test case): chứa các thông tin cần thiết để kiểm thử thành phần phần mềm theo 1 mục tiêu xác định.

* Test case gồm bộ 3 thông tin { tập dữ liệu đầu vào, thứ tự thực hiện, tập kết quả kỳ vọng}
  + Tập dữ liệu đầu vào (input): gồm các giá trị dữ liệu cần thiết để thành phần phần mềm dùng và xử lý
  + Tập kết quả kỳ vọng (output): kết quả mong muốn sau khi thành phần phần mềm xử lý dữ liệu nhập
  + Thứ tự thực hiện:



* **Ca kiểm thử** (test case): chứa các thông tin cần thiết để kiểm thử thành phần phần mềm theo 1 mục tiêu xác định.
* Test case gồm bộ 3 thông tin { tập dữ liệu đầu vào, thứ tự thực hiện, tập kết quả kỳ vọng}
  + Tập dữ liệu đầu vào (input): gồm các giá trị dữ liệu cần thiết để thành phần phần mềm dùng và xử lý
  + Tập kết quả kỳ vọng (output): kết quả mong muốn sau khi thành phần phần mềm xử lý dữ liệu nhập
  + Thứ tự thực hiện: các bước để hoàn thành ca kiểm thử từ lúc nhập dữ liệu đầu vào tới lúc nhận được kết quả đã qua xử lý của phần mềm

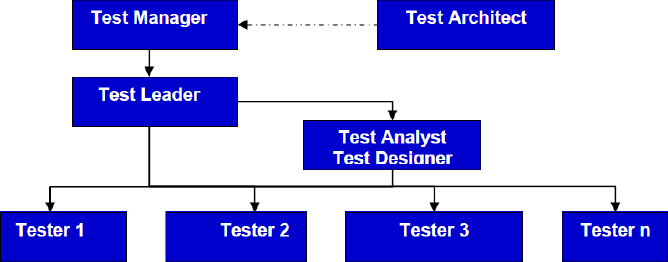


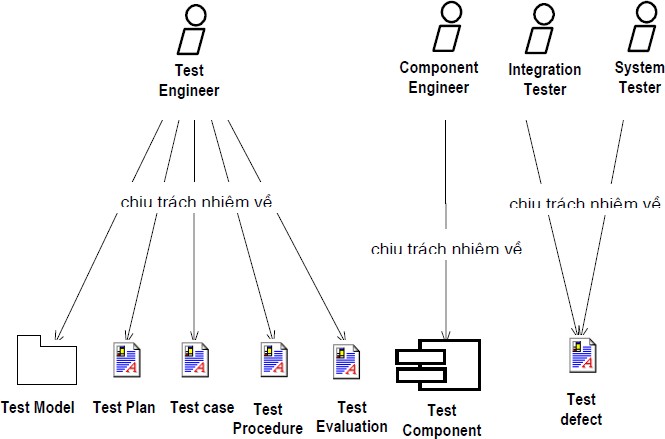
* Thiết kế các ca kiểm thử dựa trên thứ tự thực hiện các ca kiểm thử:
* **Kiểm thử nối tầng**
  + Một ca kiểm thử này có thể được xây dựng dựa trên một ca kiểm thử khác.
  + Ưu điểm của phong cách này là mỗi ca kiểm thử sẽ trở nên nhỏ hơn và đơn giản hơn.
  + Nhược điểm là nếu một ca kiểm thử sai, sẽ dẫn tới ca kiểm thử xây dựng dựa trên ca kiểm thử đó sẽ sai theo
* **Kiểm thử độc lập**
  + Mỗi ca kiểm thử được xây dựng độc lập, không dựa vào các ca kiểm thử khác, và không đòi hỏi các ca kiểm thử khác phải thực hiện thành công.
  + Ưu điểm của phong cách này là một ca kiểm thử có thể thực hiện bất cứ

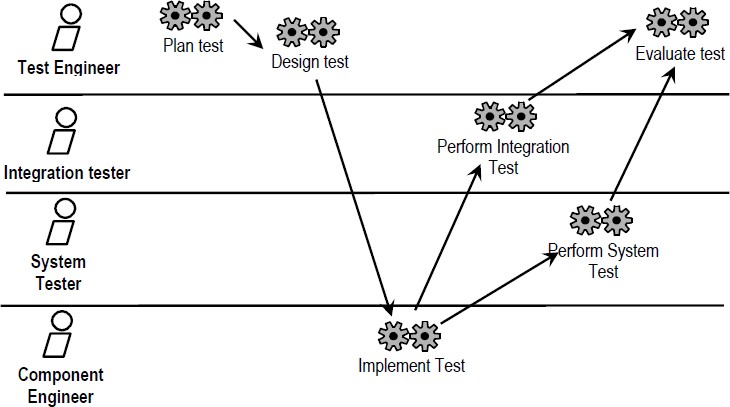
lúc nào, ko phụ thuộc vào thứ tự thực hiện các ca kiểm thử.

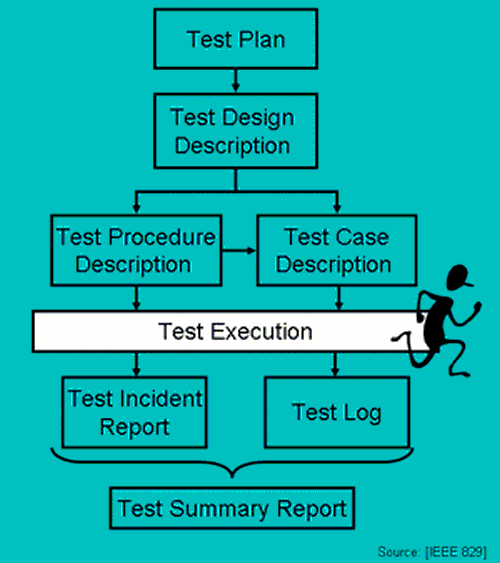
* + Nhược điểm chính là mỗi ca kiểm thử sẽ trở nên cồng kềnh và phức tạp hơn, và cũng làm cho quá trình thiết kế, thực hiện và bảo trì trở nên khó khăn hơn. 55

#### Sơ đồ tổ chức của đội kiểm thử



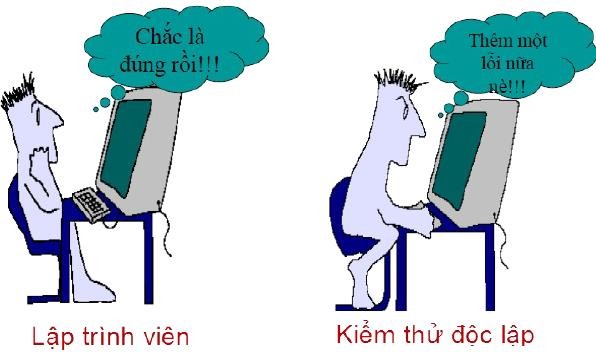




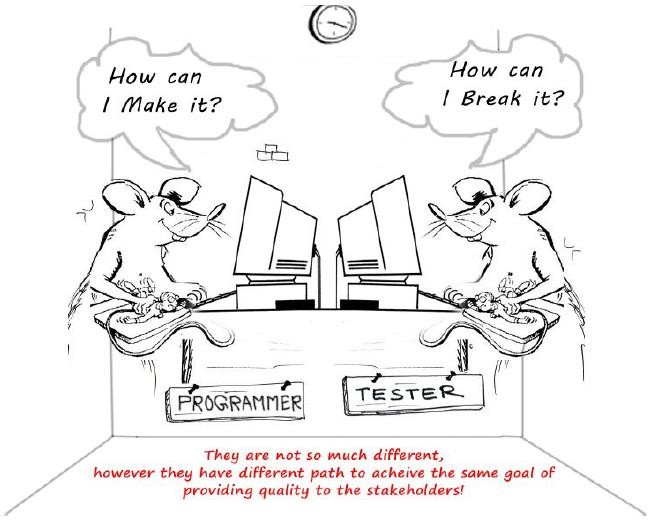




**1.9 Đối tượng thực hiện kiểm thử**







###### Chất lượng phần mềm không phải do khâu kiểm thử mà do khâu thiết kế quyết định



**1.10 Các điểm cần lưu ý khi kiểm thử**

1. Tính dễ kiểm thử phụ thuộc vào cấu trúc chương tr.nh
2. Người kiểm thử nên làm việc độc lập với người phát triển phần mềm
3. Dữ liệu thử cho kết quả bình thường thì không có ý nghĩa nhiều, cần có những dữ liệu kiểm thử để phát hiện ra lỗi
4. Khi phát sinh thêm trường hợp thử thì nên thử lại những trường hợp thử trước đó để tránh ảnh hưởng lan truyền sóng.



**1.11 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 thích hợp cho mọi phần mềm
  + Kỹ sư kiểm thử không chắc chắn họ hiểu đầy đủ về sản phẩm
  + Không có tài nguyên để thực hiện tất cả các kiểm thử
  + Không thể tìm ra được tất cả các lỗi
* **1. Lập trình viên đưa ra đoạn mã mà anh ta tin rằng không hề có lỗi.**



**CHUYỆN VUI: VÒNG ĐỜI CHẤT LƯỢNG**

* 2. Kiểm tra chất lượng sản phẩm, phát hiện 20 lỗi.
* 3. Lập trình viên sửa 10 lỗi và gửi e-mail tới phòng Thử nghiệm sản phẩm về 10

"vấn đề" còn lại mà anh ta nhất định cho rằng không phải là lỗi.

* 4. Phòng thử nghiệm sản phẩm e-mail lại rằng 5 trong số 10 đoạn sửa lỗi không hoạt động và đính kèm danh sách 15 lỗi mới.
* 5. Phòng tiếp thị gởi thông báo rằng họ đã hoàn tất khâu quảng bá cho sản phẩm. Giám đốc gọi điện xuống hỏi về tiến độ công việc và củng cố tinh thần "chiến sỹ". Phòng phát hành cử nhân viên đến nhận đĩa nguồn phần mềm. Phòng tiếp thị thông báo trên truyền hình và báo chí về việc hoãn lại ngày phát hành sản phẩm vài tuần...
* 6. Ơn trời! Cuối cùng sản phẩm cũng được phát hành.
* 7. Trong vòng một tuần, người sử dụng phát hiện ra 137 lỗi mới.
* 8. Lập trình viên phụ trách phát triển sản phẩm đã xin nghỉ phép.
* 9. Một nhóm "cứu nạn" gồm nhiều lập trình viên kỳ cựu được thành lập khẩn cấp. Sau một tuần làm việc cật lực, họ đã "thanh toán" hết 137 lỗi, nhưng lại được thông báo về 456 lỗi mới.
* 10. Mọi người tổng kết được 783 lỗi trong chương trình.
* 13. Giám đốc ngồi tại bàn giấy xem xét các báo cáo và quyết định thuê một lập trình viên mới toanh để xây dựng lại phần mềm từ đống đổ nát ban đầu.
* **1NEW. Lập trình viên mới đưa ra đoạn mã mà anh ta tin rằng không hề có lỗi.**

#### Hình thành nhóm



**Yêu cầu với Lớp.**

* + Giới thiệu thành viên nhóm

##### Tự giới thiệu thông tin cá nhân

* + - Đề xuất phương tiện truyền thông & họp nhóm
  + Đăng ký nhóm
  + Đăng ký đề tài dự án thực hiện của Nhóm trong suốt khóa học
  + Gửi danh sách tất cả các nhóm cho lớp trưởng



**Đề tài tiểu luận + báo cáo**

Mỗi nhóm sinh viên từ 2-3 người chọn 1 :

###### Đề tài 1: Hệ thống quản lý bug: Bugzilla

* Đề tài 2: Kiểm thử trên thiết bị di động (mobile testing)
* Đề tài 3: Công cụ kiểm thử tự động: Selenium
* Đề tài 4: Công cụ hỗ trợ kiểm thử tự động: Robotium.
* Đề tài 5: Công cụ hỗ trợ kiểm thử tự động: AutoIT
* Đề tài 6 : Công cụ hỗ trợ kiểm thử Mantis Bug Tracker

##### Đề tài 7: Công cụ hỗ trợ kiểm thử Sahi

* Đề tài 8 : Công cụ hỗ trợ kiểm thử Soap UI

##### Đề tài 9: Công cụ hỗ trợ kiểm thử Behavior Testing



**Test Tools**

###### Test tools:

* + Defect tracking tool
  + Test Effort tracking tool
  + Test schedule
  + Test automation tools

– *Rational Robot (Functional & Performance test)*

– *OpenSTA (Open source), Witir (Open source)*

* BT1: Viết 5 test requirements cho phần mềm Mini-bank và 4 testcases tương ứng cho mỗi test requirement. (Tuần 6)
* BT2: Thực thi kiểm thử sử dụng bộ testcase ở bài tập 1 và báo cáo kết quả. Nếu testcase failed, tiến hành report bug. (Tuần 9)
* BT3: Thực hành áp dụng các kỹ thuật hỗ trợ thiết kế testcase (white box) để thiết kế test case cho một đoạn chương trình cụ thể (java hoặc C/C++) (Tuần 12)