# PP. 345 Testing Techniques

## 1 Kỹ thuật kiểm thử hộp đen

* Phân vùng tương đương (Equivalence partition)
* Giá trị biên (Boundary values Analysis)
* Bảng quyết định (Decision table)
* Sơ đồ chuyển trạng thái (State transition)
* Cặp đôi thần kỳ (Pairwise testing)

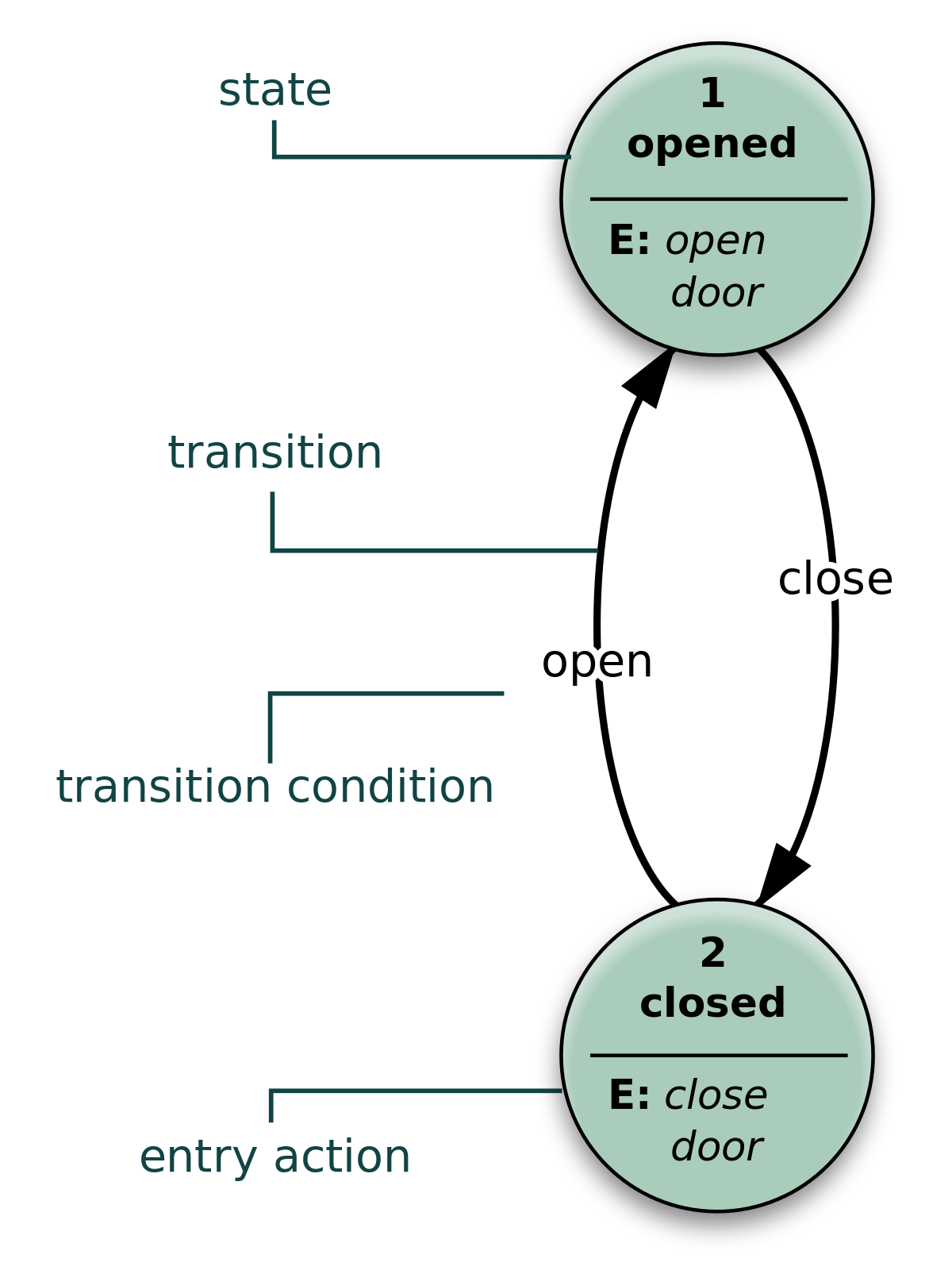
### 1.1 vùng tương đương

### 1.2 giá trị biên

### 1.3 bảng quyết định

### 1.4 Sơ đồ trạng thái

Gồm 4 phần cơ bản:

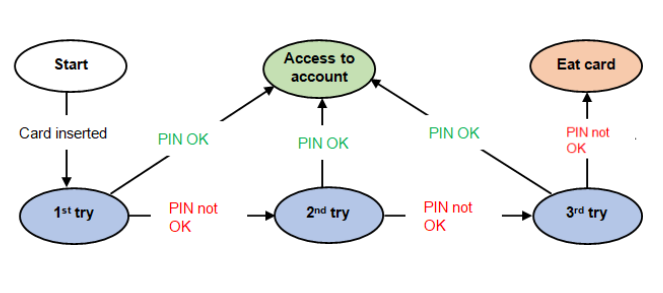
* Các trạng thái (hữu hạn) 
* Các sự dịch chuyển trạng thái
* Các sự kiện kích hoạt sự dịch chuyển trạng thái
* Các hành động là kết quả của của sự dịch chuyển

Ví dụ:

Khi NSD thực hiện nhét thẻ vào cây ATM, hệ thống sẽ hiển thị màn hình chờ nhập mã Pin. Khi NSD nhập mã PIN, hệ thống sẽ thực hiện kiểm tra:

* Lần 1: Nếu Pin hợp lệ, cho phép truy cập vào tài khoản. Nếu Pin không hợp lệ, cho phép nhập lại mã Pin
* Lần 2: Nếu Pin hợp lệ, cho phép truy cập vào tài khoản. Nếu Pin không hợp lệ, cho phép nhập lại mã Pin
* Lần 3: Nếu Pin hợp lệ, cho phép truy cập vào tài khoản. Nếu Pin không hợp lệ, nuốt thẻ.

- sơ đồ chuyển trạng thái của chức năng login trong hệ thống ATM



- testcase

|  |  |  |
| --- | --- | --- |
| **No.** | **TC** | **Expected result** |
| 1 | Insert card | Đến màn hình Wait for PIN (**1st** try) |
| 2 | Nhập PIN lần 1 đúng | Đến màn hình Access to account |
| 3 | Nhập PIN lần 1 sai | Thông báo nhập PIN sai 1 lần, cho phép nhập lại PIN |
| 4 | Nhập PIN lần 2 đúng | Đến màn hình Access to account |
| 5 | Nhập PIN lần 2 sai | Thông báo nhập PIN sai 2 lần, cho phép nhập lại PIN |
| 6 | Nhập PIN lần 3 đúng | Đến màn hình Access to account |
| 7 | Nhập PIN lần 3 sai | Thông báo nhập PIN sai 3 lần, Hệ thống nuốt thẻ |

**Ưu điểm:**

* Sơ đồ cung cấp hình ảnh tổng quan hành vi của hệ thống -> giúp tester bao quát và hiểu hành vi của hệ thống một cách hiệu quả
* Giúp tester cover được tất cả các case theo các conditions

**Nhược điểm:**

* Không áp dụng cho tất cả các hệ thống, chỉ có thể áp dụng cho các hệ thống hữu hạn trạng thái và có chuỗi các sự kiện xảy ra, kết quả của trạng thái sau phụ thuộc vào kết quả của các trạng thái trước

Bài tập thực hành 2:

Khi bạn đăng ký thành công ví điện tử thì tài khoản của bạn ở Level thường (Normal).

Từ Level Normal, nếu bạn cung cấp đủ thông tin chứng minh thư nhân dân/Hộ chiếu và thông tin thẻ ngân hàng thì tài khoản của bạn được lên Level bạc (Silver).

Hoặc từ Level Normal, nếu bạn thực hiện liên kết ngân hàng thì tài khoản của bạn cũng được lên Level Silver.

Từ Level Silver, nếu bạn thực hiện tổng các giao dịch mua sắm, tiêu dùng đạt >=5.000.000đ và <=10.000.000đ thì bạn được lên Level vàng (Gold).

Từ Level Gold, nếu bạn thực hiện tổng các giao dịch mua sắm, tiêu dùng đạt >10.000.000đ thì bạn được lên Level kim cương (Diamond).

Tuy nhiên từ Level Silver bạn cũng có thể lên Level kim cương nếu giao dịch của bạn đạt > 10.000.000đ.

Tổng gd mua > 10tr

Tổng gd mua [5tr, 10tr]

Liên kết ngân hàng

gold

silver

diamond

Normal

Cung cấp đủ in4

Tổng gd mua > 10tr

### 1.5 pairwise testing

Ví dụ: Hệ thống mua bán và bảo hành xe



- bước 1 xác định số lượng đầu vào:



- bước 2: ta xác định số lượng testcasse thử công

+ sl testcase đc tính bằng tích 2 gt lớn nhất = 3x4=12

+ ta dựng được 1 bảng gồm 12 hàng

- bước 3: thực hiện kết hợp 2 giá trị đầu vào lớn nhất



- bước 4: điền các tham số còn lại vào( chú ý điền tham số loại hàng, sao cho phủ cặp địa điểm và loại xe)



- bước 5: lặp lại bước 4, cho đến khi điền hết giá trị vào bảng



Như vậy việc testcase theo cặp , chúng ta có dòng testcase như trên

\* nhược điểm: - sl đầu vào tăng nhanh thì cũng dẫn tới sự tăng nhanh của các testcase

- vẫn có knang xảy ra lỗi khi k kết hợp toàn bộ các điều kiện đầu vào

# PP. 7 Database

## 1. data là gì?

- là thông tin của các đối tượng được lưu trữ trên máy tính

- có thể truy cập vào để truy xuất thông tin

- dl có thể lưu trữ khác nhau phù thuộc vào ngữ cảnh

- dl dc mô tả ở nhiều cách khác nhau( ký tự, ha)

## 2. cơ sở dữ liệu là gì?

- csdl là một tập hợp dl có cấu trúc liên kết với nhau và được lưu trữ trong máy tính

- csdl dc thiết kế cho user lưu trữ, truy xuất, cập nhật dl.

- cấu trúc của csdl:

+ các dl lưu trữ có cấu trúc thành các bản ghi record, các trường dữ liệu field

+ các dl lưu trữ có mối quan hệ relation với nhau

- khả năng truy xuất in4 từ csdl:

+ csdl có cấu trúc dễ dàng truy cập, quản lý và cập nhật dl

+ csdl cần dc quản lý

## 3. Hê quản trị cơ sở dữ liệu

- là phần mềm hay hệ thống được thiết kế để quản trị 1 csdl

- lưu dl, cập nhật thông tin trong 1 csdl

## 4. ngôn ngữ truy vấn

|  |  |
| --- | --- |
| **Statement** | **maining** |
| Select | Extracts data from a database |
| Update | Updates data from in a database |
| Delete | Deletes data from in a database |
| Insert into | Inserts new data into a database |

## 5. câu lệnh select

|  |  |
| --- | --- |
| **maining** | **Statement** |
| Select ra một số cột trong bảng | SELECT col1, col2,… FROM table\_name |
| Select ra môt bảng | SELECT \* FROM table\_name |
| DISTINCT: duy nhất, loại bỏ bớt bản sao | SELECT DISTINCT col1, col2,.. FROM table name |
| ORDER BY: sắp xếp theo cột nào (+ desc,asc) | SELECT \* FORM table\_name ORDER BY col1 asc |
| Where: select có điều kiện | SELECT \* FROM table\_name WHERE col1>x |
| AND: nối các kiểu kiện kết hợp | SELECT \* FROM table\_name WHERE col1>x AND col2=y |
| OR: nối các dk hoặc | SELECT \* FROM table\_name WHERE col1>x1 OR col1<x2 |
| NOT: thể hiện phủ định của mệnh đề | SELECT \* FROM table\_name WHERE NOT col1 =x |
| LIKE: giống | SELECT \* FROM table\_name WHERE col1 LIKE ‘%a%’ |
| MIN: giá trị nhỏ nhất, max, count, sum của các bản ghi | SELECT MIN(col1) where table\_name |
| AVG: giá trị trung bình của các bản ghi | SELECT AVG(col1) AS ‘name’ where table\_name |
| UNION: kết hợp kết quả của nhiều SELECT, mà không có bất kỳ bản ghi nào bị lặp  UNION ALL thì bản ghi có bản sao bị lặp | SELECT City From Customers UNION SELECT City From Customers order by City |

# PP.8 Design test case

## 1. test case là gì?

Là 1 tập hợp các điều kiện hay các giá trị, mà theo đó Tester có thể dựa vào nó để xác định xem tính năng của phần mềm có hoạt động đúng hay không.

## 2. Tại sao phải testcase?

- Lợi ích:

+ Theo dõi được tiến độ test, độ bao phủ của việc test (Track test coverage).

+ Duy trì tính nhất quán (Maintain consistency)

+ Tái sử dụng (Reusability)

+ Thuận tiện để automate

- Nhược điểm:

+ Tốn thời gian (Time-Consuming): Việc thiết kế testcase cho cả hệ thống sẽ tốn một lượng lớn thời gian và làm cho bạn chán nản.

+ Tốn nhiều thời gian để bảo trì: Khi có bất kỳ thay đổi nào trong hệ thống bạn cần sửa lại testcase của bạn.

## 3. Quy trình viết testcase



Thu thập các tài liệu -> thu thập các yêu cầu -> viết testcase

### 3.1 Collect test artifacts

- test artifacts: là các tài liệu để thiết kế test, chúng là các yêu cầu đầu vào để thiết kế testcase.

- Một vài tài liệu yêu cầu để viết testcase như: Function requirement specification(RFD), System requirement specification (SRS), UI mock up, or UI wireframe…

### 3.2 Collect requirements

- đọc từng dòng requirement

- xác định các cụm từ mô tả yêu cầu

- bỏ qua các cụm từ không phải requirement

- ghi lại các yêu cầu không rõ để hỏi lại

- ghi lại các yêu cầu vào file

Đăt 2 câu hỏi quan trọng để xem bạn đã nắm rõ yêu cầu chưa:

- yêu cầu này có rõ ràng không ?(ví dụ: bạn nắm được chức năng này làm gì không?)

- yêu cầu này có test được không? (vd: kết quả mong đợi khi test chức năng này là gì)

### 3.3 Write testcase

#### 3.3.1. Cấu trúc của một testcase

|  |  |  |
| --- | --- | --- |
| No. | Test case fields | Description |
| 1 | Test case ID | Mỗi testcase nên có 1 ID duy nhất |
| 2 | Test case Name | Mô tả ngắn gọn |
| 3 | Precondition | Yêu cầu cần được hoàn thành trước khi thực thi testcase |
| 4 | Test Procedure | Các bước để thực hiện testcase |
| 5 | Test Data | Dữ liệu thực hiện test |
| 6 | Expected Result | Kết quả thực hiện sau khi thực hiện test |

Ví dụ: testcase sign in thành công vào ứng dụng



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **PreCon** | **Test**  **Procedure** | **Test**  **Data** | **Expected**  **Result** |
| Login\_01 | Login successfully | Account Test01 is existed in system | 1. Input data:  * Username: Input correct data * Password: Input correct data  1. Click button Sign in | User name = Test01  Password= 123456 | 2. User Login system successfully: System redirect to Admin page |

## 4. cấu trúc của 1 file testcase

### 4.1 sheet: cover

- thông tin dự án

- thông tin sửa đổi và version của tài liệu

### 4.2 sheet: Test case list

- thông tin môi trường test

- Danh sách các chức năng và link với sheet test case tương ứng

### 4.3 sheet: Test case

- các test case của từng function

### 4.4 sheet: test report

- Thông tin kết quả test của từng function

- chỉ số coverage (chỉ số bao phủ)

## 5. Tổ chức các test case

### 5.1 Nhóm testcase

Chia làm 3 mức:

-group : tên chức năng lớn

- sub –group: chức năng nhỏ

- testcase: các test case của chức năng nhỏ

### 5.2 Cách chia nhóm

#### - Access right: (quyền truy cập)

+ user nào được phép truy cập

+ làm thế nào để truy cập vào màn hình

#### - UI:

+ số lượng cái item

+ vị trí, font chữ, size, màu sắc

+ trạng thái của các item (disable, enable)

+ check hiển thị giá trị default

+ check giá trị default

+ check các giá trị store trong combobox, listbox,..

#### - Validation:

+ check miền giá trị được phép input (valid partition)

+ Miền giá trị không được phép input (invalid partion)

#### - Business:

+ check nghiệp vụ, chức năng của feature

## 6. Thế nào là một testcase hiệu quả

- Test case thường có các bước đơn giản, rõ dàng và dễ hiểu.

- testcase cần hợp lệ, đơn giản và vắn tắt.

- Testcase cần truy vết được

- Testcase cần được bảo trì rõ ràng

- Sử dụng các kỹ thuật testing - test các trường hợp hợp lệ và các trường hợp không hợp lệ

- chuẩn bị dl test đa dạng: chuẩn bị dl test cho các trường hợp hợp lên và các trường hợp không hợp lệ

- cần chuẩn bị testcase cho các case phi chức năng

# PP11: **quality bug report**

## 1. Một vào thuật ngữ về lỗi phần mềm

- error / mistake (lỗi): là hành động của con người dẫn đến kết quả sai

- defect / bug / fault (lỗi, sai sót): sự hiện diện của lỗi tại thời điểm thực thi phần mềm

- failure (thất bại): sự khác biệt giữa kết quả thực tế và kết quả mong đợi

=> tóm lại: con người gây ra error, mistake trong tài liệu, code… => dẫn đến có bug, defect, fault trong phần mềm => software bị failure

## 2. các loại lỗi

1. functional bug – lỗi nghiệp vụ chức năng

2. visual – lỗi giao diện ( bị biến dạng, lệch)

3. content – lỗi đánh máy, ngữ pháp và lỗi bản địa trong văn bản

4. performance – hệ thống chậm treo

5. crash – hệ thống thoát (tắt) đột ngột

### 2.1 functional bug

Là các lỗi về luồng làm việc khi ứng dụng làm việc không theo thiết kế. Các lỗi này gây nên sự khác biệt giữa kết quả thực tế và kết quả mong đợi của hệ thống.

Vd:

- link bị lỗi,

- search and filters trả ra kết quả sai

- button không hoạt động

- kết quả tính toán bị sai

### 2.2 visual

Các lỗi ảnh hưởng đến giao diện như thiếu element hoặc ảnh trên môt trang

Vd:

- các element hoặc nội dung chưa được căn chỉnh

- content bị tràn ra ngoài khung

- màu sắc trên một link, button hoặc menu không đồng nhất

- thiếu ảnh

### 2.3 content

Các lỗi nội dung ảnh hưởng đến văn bản của một trang như: đánh vần, ngữ pháp, và lỗi bản địa

Vd:

- lỗi bản địa: từ sai được sử dụng

- spelling and capitalization error such as uTEuT

- lỗi đánh vần viết hoa: devPrO

- dấu câu được sử dụng sai trong văn bản (. , : ; ‘ “)

### 2.4 Performance

Các tính năng mất nhiều thời gian để tải hơn mức cần thiết hoặc điều hướng chậm trong ứng dụng

Ví dụ:

- ứng dụng phản ứng chậm khi điều hướng trong suốt các tính năng

- ứng dụng hoặc trang mất quá nhiều thời gian để tải

- ứng dụng đóng băng hoặc không phản hồi

### 2.5 Crash

ứng dụng bị thoát hoặc đóng lại không như mong đợi khi sử dụng các tính năng

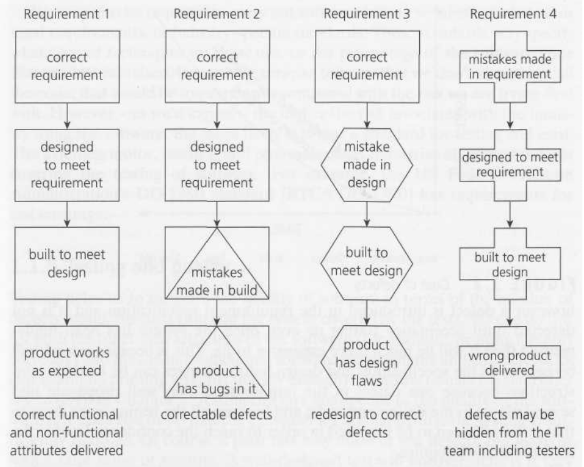
ví dụ:

- trang web bị treo và không phản hồi, cuối cùng dẫn đến lỗi hoặc đóng trình duyệt

- máy tính - ứng dụng đóng băng thiết bị, treo thời gian dài hoặc đóng băng đột ngột

- điện thoại di động - ứng dụng đóng đột ngột do lỗi

3. Nguồn gốc và cách khắc phục lỗi



- requirement 1: yêu cầu chính xác -> thiết kế đáp ứng yêu cầu ->xây dựng để đáp ứng thiết kế ->sản phẩm làm việc như mong đợi

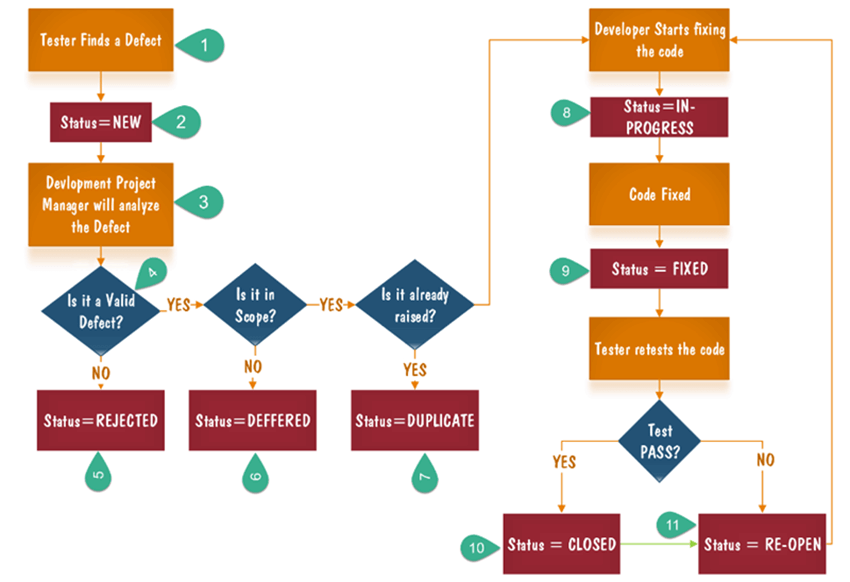
->các thuộc tính của functional và non-functional chính xác và được bàn giao

- requirement 2:yêu cầu chính xác -> thiết kế đáp ứng yêu cầu -> sai lầm trong xây dựng -> sản phẩm có lỗi trong đó

- requirement 3: yêu cầu chính xác -> sai lầm trong thiết kế -> xây dựng để đáp ứng thiết kế -> sản phẩm có lỗ hổng thiết kế

- requirement 4: sai lầm trong yêu cầu ->thiết kế đáp ứng yêu cầu -> xây dựng để đáp ứng thiết kế -> sai sản phẩm bàn giao

## 3. vòng đời của bug



1. tester tìm ra defect

2. chuyển trạng thái: status = new

3. dev, tester cùng phân tích defect

4. xem nó có phải là lỗi hợp lệ không ( khi k tìm dc tiếng nói chung thì testlead và PM sẽ giải đưa ra kết luận)?

5. (No) chuyển trạng thái từ chối: status= rejected

6. (Yes) nó có nằm trong phạm vi của dự án không?

7. (No) chuyển trạng thái hoãn lại: status = deffered

8. (Yes) nó đã xuất hiện trước đó chưa

9. (Yes) chuyển trạng thái là nhân đôi: status= duplicate

10.(No) dev bắt đầu sửa, chuyển trạng thái là trong tiến trình: status= in progress

11. dev fix xong, chuyển trạng thái là đã fix: status = fixed

12. tester tiến hành kiểm thử lại code

13. xem code đã sửa đã pass hay chưa?

14.( Yes) đã pass, chuyển trạng thái là đã đóng: status= closed

15. (no) không pass, chuyển trạng thái là mở lại: status= re-open (chuyển đến bước 10 và tiến hành sửa lại)

## 4. Nội dung log bug

Các thông tin của 1 bug report:

Report khác:

- date:

- assigned to:

- status

- title

- description/ summary:

- enviroments

- step to reproduce:

1. 2. 3. …

- Actual results:

- expected results:

- severity: mức độ nghiêm trọng

- priority level: độ ưu tiên

- The title

- Precondition

- Procedure/ Step

- Expected result

- Actual result

- Attachment

### 4.1 cách viết title

- title là bộ mặt của bug report. Một title tốt có thể chứa nội dung tổng quát của 1 bug

- cần tránh tiêu đề chung chung.

Vd: chức năng A hoạt động không tốt/ không hoạt động, có vấn đề với chức năng B, chức năng C bị lỗi

- hãy mô tả chức năng A not working như thế nào:

+ thay vì viết : function create account does not working

=> hãy viết: display error message on clicking the save button while creating a new user, hence unable to create a new user in the application.

+ thay vì viết: có vấn đề với GUI

=> hãy viết: GUI bị sai font chữ

### 4.2 cách viết procedure/ steps

- procedure / step là phần thân của bug report

- mục đích: giúp người đọc tái hiện ( procedure) được bug

- chú ý: nên đánh số cho từng step để dev có thể dễ dàng có thể tái hiện lại

Vd:

1. login into the application

2. navigate to users menu > new user

3. filled all user information fields

- username =

4. clicked on the “save” button

### 4.3 Cách viết expected

- expected result: là kết quả mong đợi trước khi thực hiện các steps của phần procedure

- chú ý: expected result nên đánh số thứ tự tương ứng theo các bước thực hiện

Vd:

4. a success message “ new user has been created successfully” should be shown

### 4.4 cách viết actual result

- actual result: là kết quả thực hiện các steps của phần procedure

- chú ý: Actual nên đánh số thứ tự tương đương theo các bước thực hiện

Ví dụ

4. display error message: “ error creating account”

### 4.5 attachment

- nên chụp màn hình lỗi khi thực hiện test để làm evidence (bằng trứng) cho dev

# cuối trang