Môn học

KIỂM THỬ PHẦN MỀM

Chương I

TỔNG QUAN

Nội dung

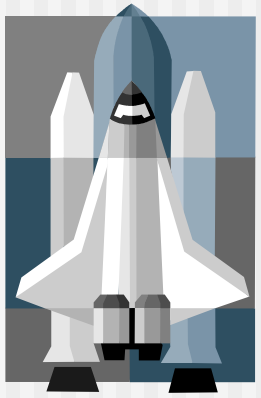
1. Tại sao kiểm thử lại cần thiết?
2. Khái niệm và mục tiêu kiểm thử phần mềm
3. Các nguyên tắc kiểm thử cơ bản
4. Quy trình kiểm thử phần mềm

I.1Tại sao kiểm thử lại cần thiết?

- Ngữ cảnh hệ thống phần mềm
- Nguyên nhân xuất hiện các khiếm khuyết
- Kiểm thử và chất lượng
- Kiểm thử bao nhiêu là đủ?

Ngữ cảnh hệ thống phần mềm

- Phần mềm xuất hiện trong hầu hết các lĩnh vực như ngân hàng, mua sắm, giáo dục, y tế, hàng không,…
- Hầu hết người dùng đều trải qua tình huống phần mềm làm việc không như mong đợi nghèo nàn về giao diện, chức năng không phù hợp
- Mỗi phần mềm có mức độ rủi ro và ảnh hưởng khác nhau ảnh hưởng về tính mạng (máy bay, y tế)



Ngữ cảnh hệ thống phần mềm

- Thị trường phần mềm
  - Ngày càng đa dạng, cạnh tranh lớn, lượng người dùng tăng cao

→ Kiểm thử có ý nghĩa quan trọng đối với sự thành công của các sản phẩm phần mềm

Software Error, Fault, và Failure

a/b => người dùng nhập
b = 0 => lỗi runtime

thiếu dấu ; => lỗi biên dịch

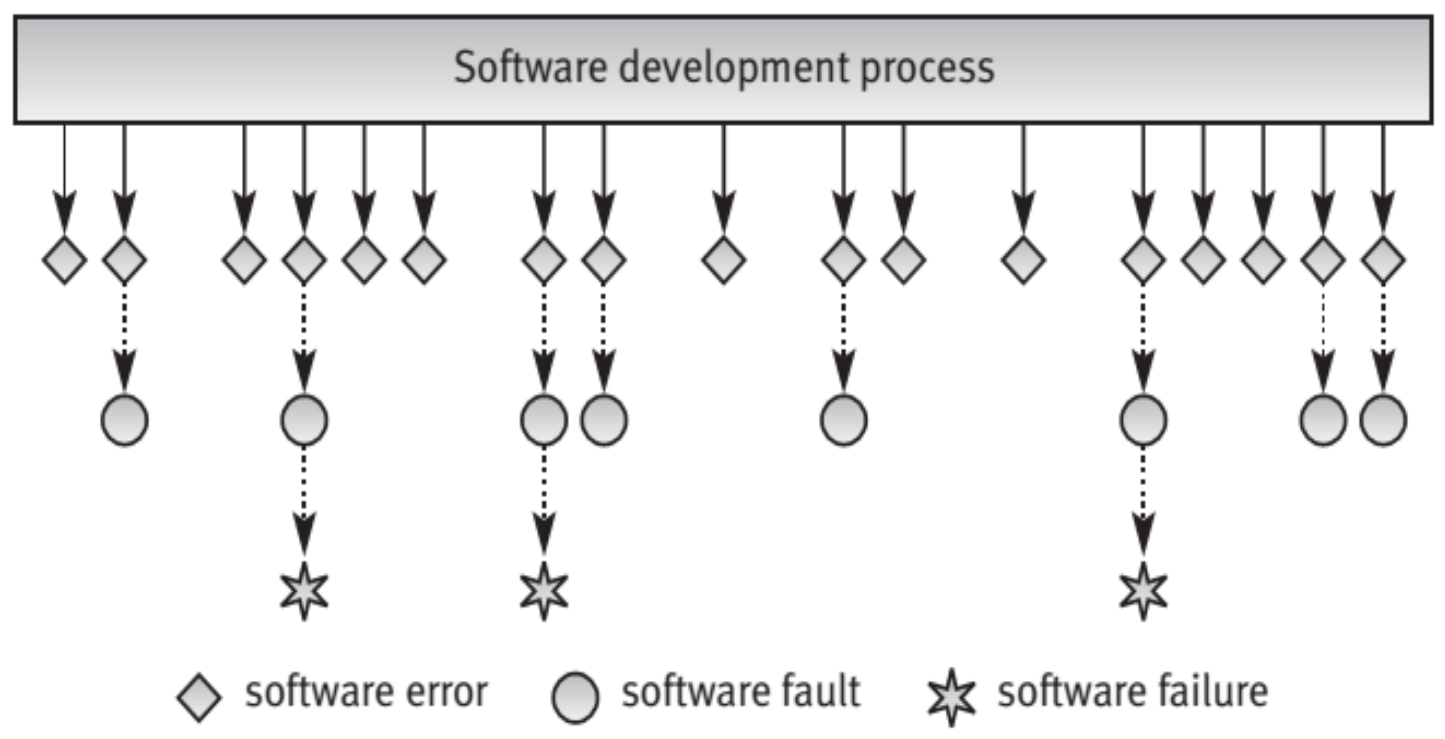
- Error/Mistake: hành động của con người gây ra kết quả không chính xác
- Fault/Bug/Defect: kết quả của error trong PM
- Failure: hệ thống thực hiện không mong đợi. Fault gây ra failure trong PM.

Thất bại của phần mềm: phần mềm chạy k được, đang chạy bị đứng, treo
có một số pm chạy được nhưng không đáp ứng được NHU CẦU CỦA KH => cũng failure
Thất bại PM là hệ quả của bug (bug tiềm ẩn). tuy nhiên không phải mọi bug đều xảy ra failure

vd: nhập a, b khác 0 => tùy theo điều kiện kích hoạt trường hợp failure.

Software Error, Fault, và Failure

Minh họa mối quan hệ giữa Error, Fault và Failure



Example

- Bệnh nhân đưa ra một loạt các triệu chứng cho bác sĩ – Failures
- Bác sĩ chẩn đoán nguyên nhân (bệnh tật) – Fault
- Bác sĩ tìm thấy các tình trạng bất thường bên trong (huyết áp cao, nhịp tim không đều, vi khuẩn trong máu,…) - Errors

cần thiết kế bộ test làm sao để lộ ra được bug

Example

```
int numZero (int *arr, int size)
{
   int count = 0;
   for (int i = 1; i < size; i++)
   {
      if (arr [ i ] == 0)
      {
         count++;
      }
   }
   return count;
}
```

đếm có bao nhiêu phần tử 0 trong mảng

Fault: Bỏ sót tìm kiếm tại vị trí đầu tiên của mảng

Test 1
[2, 7, 0]
Expected: 1
Actual: 1

Error: gán i = 1
Failure: không có

Test 2
[0, 2, 7]
Expected: 1
Actual: 0

Error: gán i = 1 → biến count nhận giá trị sai
Failure: giá trị count trả về là 0

Thất bại phần mềm

- Máy bức xạ THERAC-25: do kiểm thử kém về độ an toàn của phần mềm khiến 3 bệnh nhân thiệt mạng
- Ariane 5: tên lửa gặp một "bug" khiến các kĩ sư phải nhấn nút hủy nhiều lần vì phần mềm đang cố lưu 1 số 64-bit vào bộ nhớ 16-bit → mất khoảng 370 triệu đô.

=> tràn số - vượt quá phạm vi lưu trữ của kiểu dữ liệu

Thất bại phần mềm

- Bitcoin Hack, Mt. Gox: sàn giao dịch Bitcoin của Nhật bị hack năm 2011 → mất 850.000 bitcoins vì lỗ hổng trong hệ thống phần mềm

Testers nên cố gắng tìm ra khiếm khuyết trước khi các khiếm khuyết tìm đến người dùng

Thảo luận

- Những nguyên nhân nào gây ra khiếm khuyết trong PM?

Nguyên nhân gây ra khiếm khuyết trong PM

NN chủ quan

- Xác định sai yêu cầu (kể cả thiếu, dư yêu cầu)
- Nhầm lẫn khi giao tiếp giữa developer và client
- Cố ý sai lệch so với yêu cầu PM
- Sai trong logic thiết kế

cố ý sai lệch: tái sử dụng mà không chỉnh sửa lại cho phù hợp với phần mềm hiện tại; bỏ bớt một số chức năng của KH (do cận deadline, hết ngân sách...)

Nguyên nhân gây ra khiếm khuyết trong PM

- Sai sót trong quá trình coding
- Vi phạm chuẩn, mẫu chung
- Sai sót trong quá trình kiểm thử
- Tài liệu viết không chính xác hoặc không đầy đủ

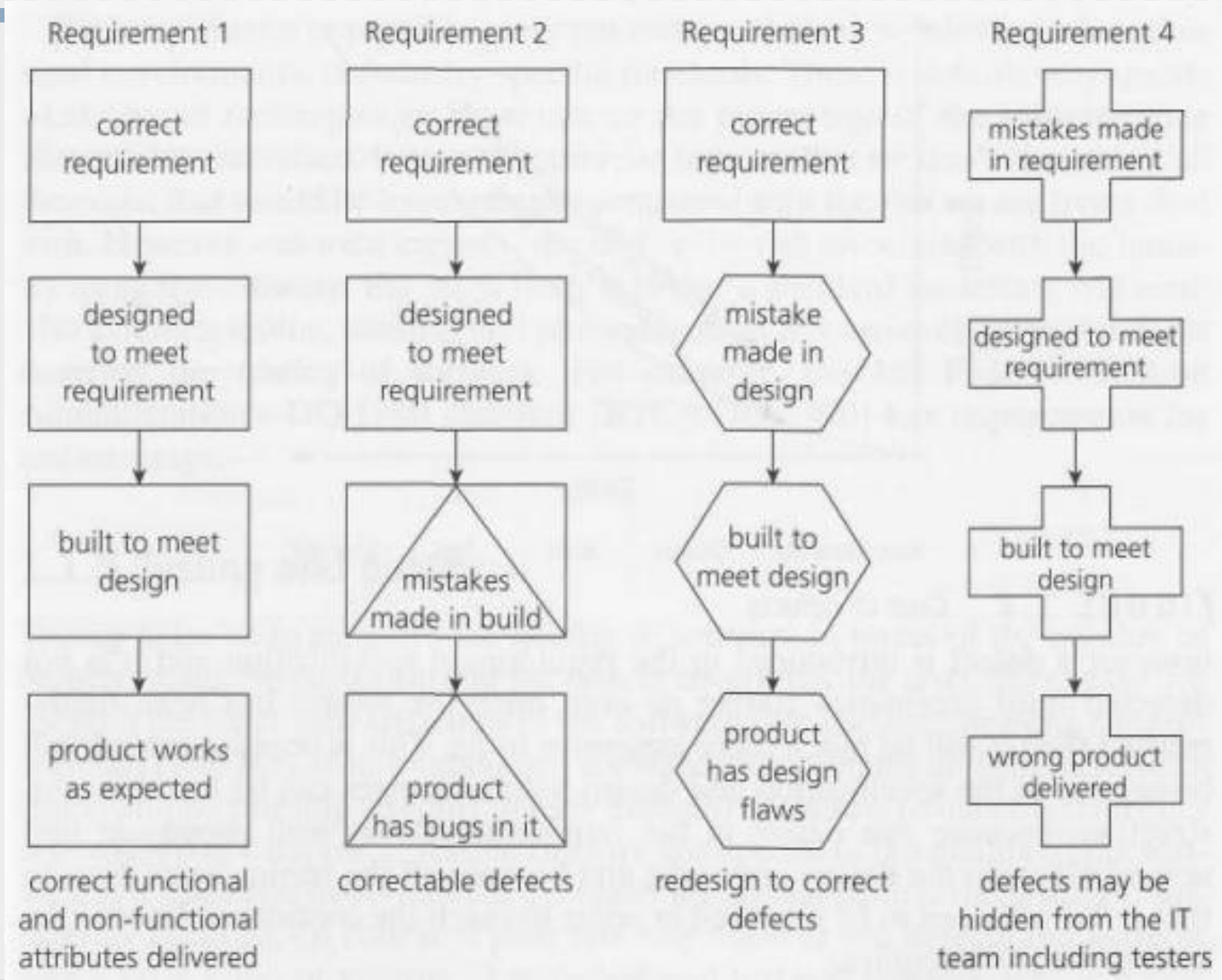
Nguyên nhân gây ra khiếm khuyết trong PM

NN Khách quan

- Lỗi khi sử dụng phần mềm (do user không nắm quy trình sử dụng)
- Điều kiện môi trường, công nghệ thay đổi (PM không tương thích với hệ điều hành)
- Tương tác nhiều hệ thống
- Cố ý gây thiệt hại (virus, malware,…)
- …

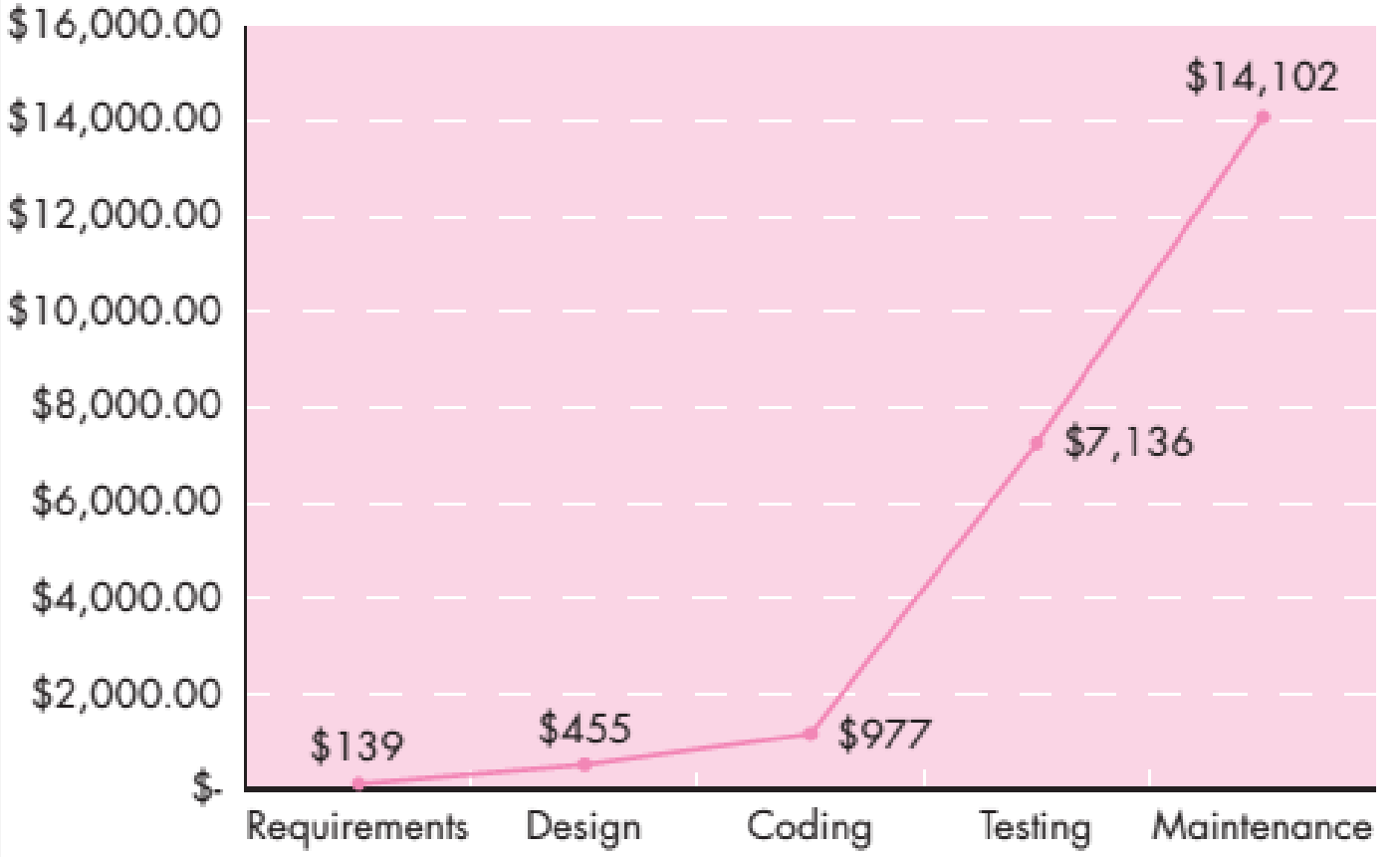
trường hợp nào nghiêm trọng nhất? 4 - sai dây chuyền => Phân tích xác định yêu cầu là bước quan trọng nhất

Phát sinh các khiếm khuyết



Chi phí tìm và sửa các khiếm khuyết

kiểm thử phần mềm thực hiện càng sớm càng tốt, khi phân tích xđ yêu cầu



Kiểm thử và chất lượng

- Kiểm thử giúp đo lường chất lượng PM
- Kiểm thử có thể tìm ra lỗi, khi lỗi được fix thì chất lượng PM được cải thiện
- Kiểm thử sẽ thực hiện test:
  - Chức năng của hệ thống
  - Yêu cầu phi chức năng

→Tại sao cần phải kiểm thử?

- Phần mềm luôn luôn có lỗi
- Thất bại phần mềm sẽ rất đắt
- Giúp kiểm tra về độ tin cậy của PM
- Đánh giá và quản lí rủi ro

I.2 Khái niệm và mục tiêu kiểm thử phần mềm

- **Kiểm thử là gì?**

  Kiểm thử là quá trình vận hành chương trình để tìm ra lỗi trước khi chuyển giao cho người dùng cuối

  (Glen Myers)

Test Case (TC)

là một trường hợp kiểm thử

- Gồm 3 thành phần chính:
  - Các bước thực hiện test
  - Dữ liệu test (input)
  - Kết quả mong đợi (không có KQ thực tế - khi chạy test case đó trên phần mềm)
- Một test case tốt: => làm lộ ra ít nhất 1 lỗi trong chương trình
  - Hiệu quả
  - Tổng quát
  - Dễ cập nhật, sữa chữa
  - Kinh tế

Lợi ích của quá trình kiểm thử

- Cải tiến chất lượng phần mềm
- Giảm chi phí
- Duy trì sự hài lòng của khách hàng
- Giúp cải tiến quy trình phát triển thông qua việc rà soát các khiếm khuyết và thất bại

Kiểm thử triệt để (Exhaustive testing)

- Thực hiện tổ hợp tất các đầu vào có thể có và điều kiện tiên quyết.
- Cần thực hiện một số lượng lớn các Test case → Chiếm thời gian rất lớn, không thực tế.

nhập số nguyên từ 1-100: 3 test case:
1. 1 giá trị nằm trong đoạn từ 1-100 => nếu đúng thì các số đều đúng
2. -1 < 1
3. 101 > 100

Kiểm thử bao nhiêu là đủ?

PM quá nhiều lỗi => quá tệ => dừng lại, làm lại từ đầu

- Phụ thuộc vào kết quả test
- Phụ thuộc vào các rủi ro:
  - Bỏ lỡ các faults quan trọng
  - Phát sinh chi phí thất bại
  - Phát hành phần mềm chưa được test
  - Thực thi các test không hiệu quả
  - Mất uy tín, thị phần
  - …

Kiểm thử bao nhiêu là đủ?

- Sử dụng các rủi ro để quyết định
  - Cái gì cần test trước
  - Cái gì cần nhấn mạnh
  - Cái gì chưa cần test trong thời điểm hiện tại
  - Phân bổ thời gian cho việc kiểm thử

  → Thực hiện sắp xếp độ ưu tiên cho các TCs

=> Lý do: TC nào có độ ưu tiên cao hơn sẽ thực thi trước => phát hiện trước
Bất kỳ thời điểm nào dừng việc test (do thời gian, ngân sách) => đã test các trường hợp quan trọng trước, an tâm hơn
độ ưu tiên cao gắn liền với việc quan trọng
vậy cái nào quan trọng? tùy vào chức năng, nếu lỗi xảy ra thì ảnh hưởng nhiều hay ít với khách hàng

Sắp xếp độ ưu tiên cho các Test cases

- Test tại vùng có thể gây thất bại nghiêm trọng nhất
- Test tại vùng có khả năng xảy ra lỗi nhất
- Test các yêu cầu đặc biệt quan trọng đối với khách hàng
- Test vùng thường xảy ra thay đổi
- Chú trọng vùng phức tạp về mặt kỹ thuật

Mục tiêu kiểm thử phần mềm

=> không nên đặt mục tiêu là free bug, vì nó phi thực tế.

- Phát hiện ra nhiều lỗi nhất có thể và sớm nhất có thể
- Có một PM đạt được chất lượng ở mức có thể chấp nhận
- Thực hiện các tests cần thiết và hiệu quả trong phạm vi ngân sách và lịch biểu đã đưa ra.

Gỡ lỗi để loại bỏ "Bug"

- Gỡ lỗi (Debugging)
  - Là quá trình lập trình viên định vị bug trong code, fix code, và kiểm tra lại code đã thực thi đúng chưa.

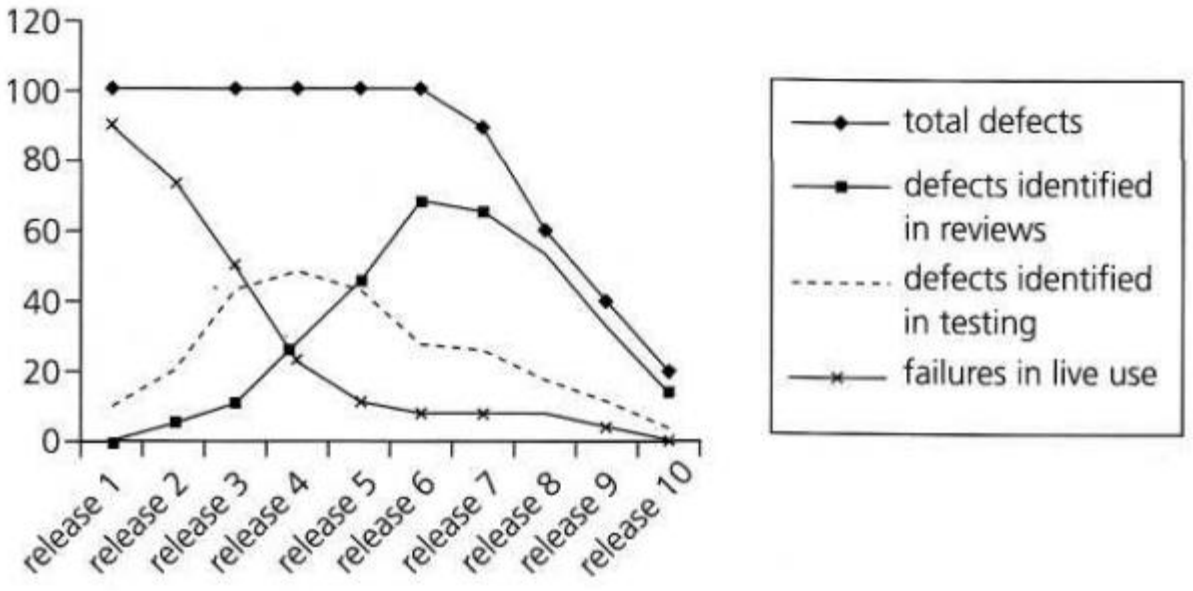


FIGURE 1.3 Changes in defect numbers during process improvement

I.3 Các nguyên tắc kiểm thử cơ bản

1. Kiểm thử có thể chỉ ra sự hiện diện của "bug", nhưng không chứng minh được PM không có "bug".
2. Kiểm thử triệt để là không thể.
3. Các hoạt động kiểm thử nên bắt đầu sớm nhất có thể
4. Các "bug" có xu hướng phân cụm

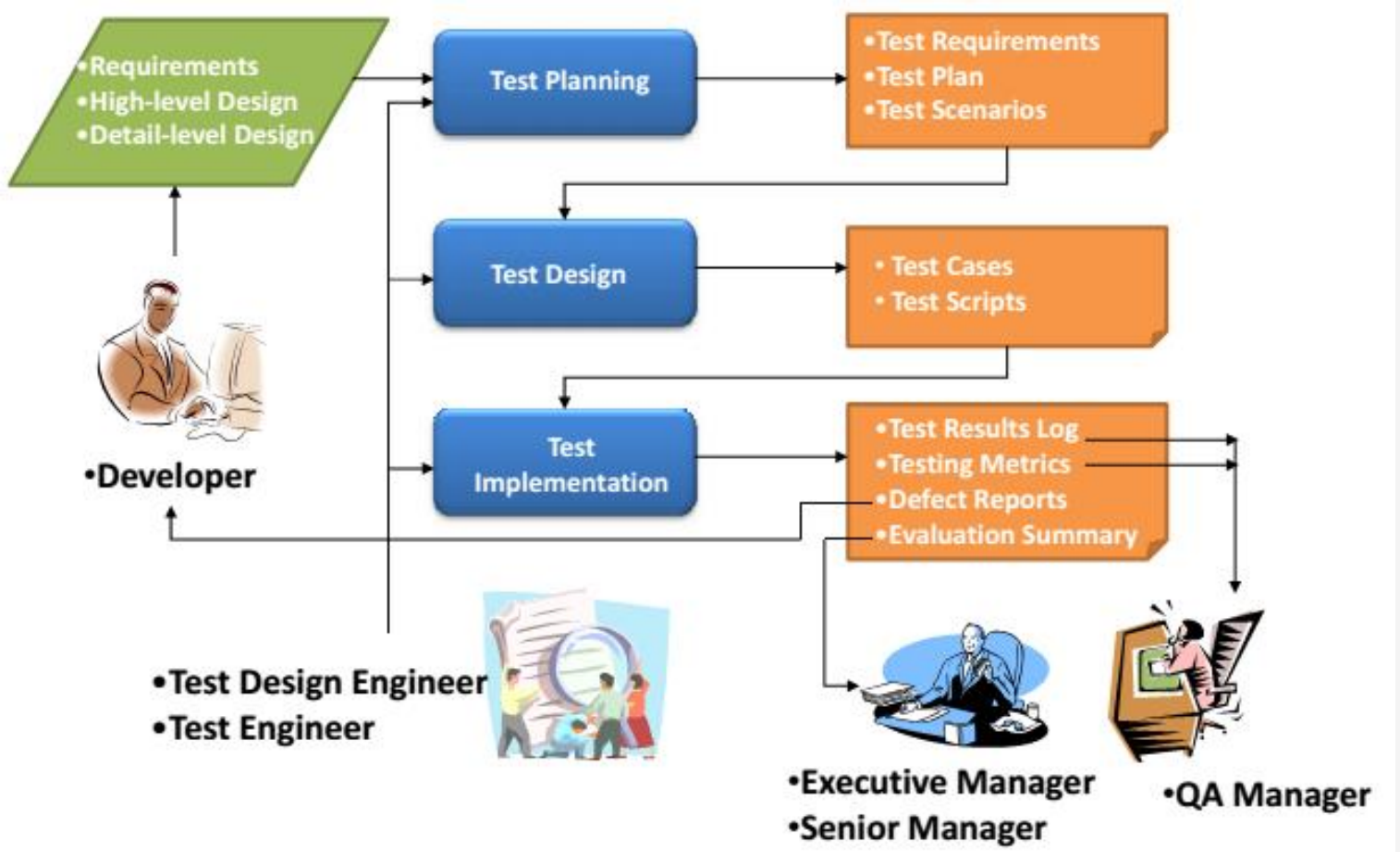
I.3 Các nguyên tắc kiểm thử cơ bản

5
- Các test cases cần được rà soát, sửa đổi, và thực hiện ở các phần khác nhau của PM để tìm ra nhiều "bug" tìm ẩn

6
- Kiểm thử phụ thuộc vào ngữ cảnh

7
- Tìm và fix "bug" sẽ không hữu ích nếu PM không sử dụng được, không đáp ứng nhu cầu và mong đợi của người dùng.

I.4 Quy trình kiểm thử cơ bản



I.4 Quy trình kiểm thử cơ bản

- Bao gồm các hoạt động: có quy trình, theo vòng lặp
  - Lập kế hoạch và kiểm soát quá trình test
  - Phân tích xác định các điều kiện, thiết kế TCs (do tester làm)
  - Thực thi tests (do tester làm - run chương trình, nhập test case và quan sát KQ)
  - Đánh giá các tiêu chuẩn và báo cáo
  - Kiểm tra hoàn thành tests (quá trình kiểm thử đã kết thúc được chưa? dựa vào tiêu chí đã đưa ra trước đó).

kiểm thử cn nào, ai tham gia, sd pp nào, sử dụng công cụ nào tg kiểm thử là bao nhiêu...

do leader làm, có thể tester hỗ trợ

Bước 1: Vạch kế hoạch

- Thiết lập chiến lược test
- Xác định những người tham gia: testers, QA, development, support,…
- Xác định các yêu cầu, đặc tả về chức năng
- Chuẩn bị cơ sở hạ tầng cho test
- Lên lịch biểu cho các hoạt động test
- Xác định phạm vi và các rủi ro kiểm thử
- Xác định tiêu chí hoàn thành test
- …

Kiểm soát quá trình test

- Đo lường và phân tích các kết quả rà soát, kiểm thử
- Theo dõi và lập hồ sơ tiến độ, độ bao phủ và tiêu chí hoàn thành test
- Thực hiện các hoạt động: thắt chặt tiêu chí hoàn thành hoặc thay đổi ưu tiên fix các “bug”
- Đưa ra quyết định

Bước 2: Phân tích, thiết kế test

- Xem xét cơ cở test
- Xác định các điều kiện test và độ ưu tiên của nó sử dụng các kỹ thuật test
- Thiết kế các Test cases
- Triển khai các Test cases: Chuẩn bị kịch bản (thủ tục) và dữ liệu test, tạo test suites
- Cài đặt môi trường test

Bước 3: Thực thi tests

1. manual: thực hiện bằng tay
2. automatic: thực hiện công cụ

- Thực thi test suites và các TCs riêng lẻ đã được thiết kế dựa theo script:
  - Thực hiện trước các TCs quan trọng nhất
  - Có thể thực thi bằng tay hoặc tự động
  - Không thực thi tất cả các TCs nếu:
    - Có quá nhiều lỗi ở các TCs trước
    - Áp lực thời gian

Bước 4: Đánh giá và báo cáo

- Ghi lại các phiên bản của PM trong khi test
- Với mỗi TCs, ghi lại kết quả thực tế, độ bao phủ test
- So sánh kết quả thực tế với kết quả mong đợi
- Sau khi lỗi được fix thì cần phải test lại
- Viết báo cáo trạng thái kiểm thử (bug report)

sau khi viết báo cáo, đưa cho leader để kiểm tra lại đó có thực sự lỗi hay không, tránh các quan điểm chủ quan.
sau khi KT leader sẽ đưa cho dev để fix.
sau khi dev fix, báo lại cho tester, rồi tester sẽ kiểm thử lại xem bug đó đã sửa hay chưa.

Bước 5:
Kiểm tra hoàn thành tests

- Tiêu chuẩn kết thúc test đã được đưa vào test plan
- Nếu các tiêu chuẩn chưa được đáp ứng thì thực hiện lại từ việc thiết kế thêm các TCs
- Các tiêu chuẩn có thể ứng dụng với tất cả các mức test
  - Độ bao phủ
  - Các lỗi được tìm thấy
  - Chi phí hoặc thời gian

Khi đến bước 5, các tiêu chí, vd độ bao phủ trên chưa thỏa mãn, ta phải quay lại **bước 2**: bổ sung tc -> 3 -> 4 -> 5