KIỂM THỬ PHẦN MỀM

KIỂM THỬ HỘP ĐEN

ThS. Dương Hữu Thành
Khoa CNTT, Đại học Mở Tp.HCM
thanh.dh@ou.edu.vn

Nội dung chính

1. Tổng quan kiểm thử hộp đen.
2. Quy trình kiểm thử hộp đen.
3. Các loại kiểm thử hộp đen
4. Các kỹ thuật kiểm thử hộp đen.
   - Phân vùng tương đương.
   - Phân tích giá trị biên.
   - Bảng quyết định.
   - Dịch chuyển trạng thái.
   - Kiểm thử dựa trên lược đồ use case.
   - Kiểm thử dựa trên kinh nghiệm.

Tổng quan kiểm thử hộp đen

Kiểm thử hộp đen (black-box testing) còn gọi kiểm thử **dựa trên đặc tả** (specification-based testing) vì thông tin duy nhất làm cơ sở để kiểm thử hộp đen là bảng đặc tả yêu cầu chức năng của từng thành phần phần mềm.

Tester tập trung vào những gì phần mềm làm được (**WHAT**), không quan tâm nó làm như thế nào (HOW).

Tổng quan kiểm thử hộp đen

Kiểm thử viên sẽ **tương tác với giao diện** người dùng, **cung cấp dữ liệu đầu vào**, **kiểm tra dữ liệu đầu ra**, mà không cần quan tâm mã nguồn, không cần biết chi tiết các công việc bên trong, cách thức và nơi nào xử lý dữ liệu đầu vào đó

Tổng quan kiểm thử hộp đen

Ưu điểm

- Không cần truy cập mã nguồn.
- Tách biệt khung nhìn của user và developer.
- Nhiều người có thể tham gia test.

Khuyết điểm

- Kiểm thử không có hiệu quả cao.
- Khó thiết kế test case.
- Khó kiểm thử phủ được hết các trường hợp.
- Không có định hướng kiểm thử rõ ràng.

Quy trình kiểm thử hộp đen

Phân tích các đặc tả chức năng của các thành phần phần mềm.

Thiết kế test case để kiểm thử.

Thực thi các test case để kiểm thử.

So sánh kết quả đạt được với kết quả mong muốn trong từng test case.

Lập báo cáo kết quả kiểm thử.

Các loại kiểm thử hộp đen

Kiểm thử chức năng (functional testing) kiểm thử các chức năng chỉ định của hệ thống, tính an toàn của hệ thống (security testing), kiểm thử sự tương tác của hệ thống với các thành phần khác được chỉ định

Kiểm thử phi chức năng (non-functional testing) tập trung kiểm thử các khía cạnh hành vi của hệ thống như tính tiện dụng, tính khả chuyển, hiệu năng hệ thống như load test hay stress test

Các loại kiểm thử hộp đen

Kiểm thử có sự thay đổi:

Sau khi lỗi tìm thấy được giải quyết thì hệ thống cần được kiểm tra lại để xác nhận vấn đề đã thực sự được giải quyết, kiểm thử này gọi là **retesting** hoặc **confirmation testing**.

Khi mã nguồn hệ thống có thay đổi hoặc môi trường hệ thống thay đổi, ta cũng cần thực thi lại một tập các test case đã thiết kế trước đó, để đảm bảo không có lỗi mới phát sinh từ sự thay đổi đó, kiểm thử này gọi là kiểm thử hồi quy (**regression testing**).

Các kỹ thuật kiểm thử hộp đen

Phân vùng tương đương (Equivalence Partitioning hoặc Equivalence Class)

Phân tích giá trị biên (Boundary Value Analysis)

Bảng quyết định (Decision Tables hay còn gọi là Cause Effect)

Dịch chuyển trạng thái (State Transition Testing)

Sử dụng use case (Use case Testing)

Các kỹ thuật kiểm thử hộp đen

Kỹ thuật phân vùng tương đương và phân tích giá trị biên thường được áp dụng cho các tình huống các **đầu vào cụ thể** và **ít có quan hệ ràng buộc nhau**.

Kỹ thuật bảng quyết định và dịch chuyển trạng thái tập trung vào **logic** và **quy tắc (rule) nghiệp vụ**.

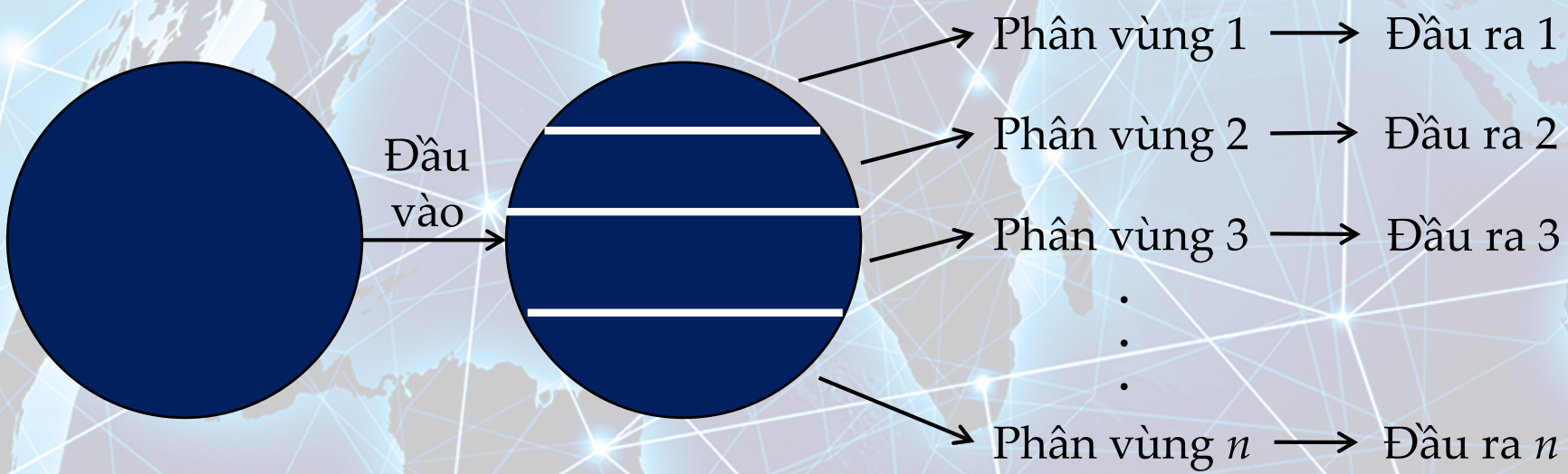
Kiểm thử phần mềm

PHÂN VÙNG TƯƠNG ĐƯƠNG
(Equivalence Partitioning)

Phân vùng tương đương

Ý tưởng của kỹ thuật này nếu một **giá trị đại diện** trong nhóm đúng thì các giá trị còn lại trong nhóm cũng đúng và ngược lại.

Phương pháp phù hợp cho các bài toán có giá trị đầu vào là một miền xác định.



Phân vùng tương đương

Phân vùng tương đương (EP) là phân chia một tập các điều kiện kiểm thử thành các tập con có các giá trị tương đương nhau và kiểm thử các tập con này.

Mục đích của EP là **giảm thiểu số lượng test case** không cần thiết khi kiểm thử.

EP có thể áp dụng tất cả mức độ kiểm thử.

Phân vùng tương đương

Hai giá trị tương đương theo một trong các cách sau:

- Chúng tương tự nhau (intuitive similarity).
- Tài liệu đặc tả mô tả chương trình sẽ xứ lý chúng theo cùng một cách thức (specified as equivalent)
- Chúng lái chương trình theo cùng đường (chẳng hạn cùng nhánh if) (equivalent path).
- Chúng cho cùng kết quả với những giả thiết đưa ra.

Ví dụ

Xếp loại cuối năm học sinh ở trường THPT dựa trên điểm trung bình (ĐTB) như sau:

0 ≤ ĐTB < 5: yếu, kém

5 ≤ ĐTB < 7: trung bình

7 ≤ ĐTB < 8: khá

8 ≤ ĐTB < 9: giỏi

9 <= ĐTB <= 10: xuất sắc

Biết điểm trung bình làm tròn 1 chữ số thập phân.

Ví dụ

Để kiểm thử ứng dụng này chia thành 7 phân vùng tương đương, trong đó:

5 phân vùng hợp lệ (valid): [0, 5), [5, 7), [7, 8), [8, 9), [9, 10].

2 phân vùng không hợp lệ (invalid): < 0 và > 10

Ví dụ

Tester không nên chỉ test những gì trong đặc tả yêu cầu, mà còn phải **nghĩ ra những thứ không được đề cập**. Trong ví dụ trên 2 phần vùng invalid không được đề cập trong đặc tả, nhưng cần được kiểm thử.

Khi thiết kế test case phải **đảm bảo tất cả các phân vùng (valid & invalid) được test** qua ít nhất một lần. Trong ví dụ trên ít nhất cần test 7 điểm trung bình sau đại diện với 7 phân vùng - 5.0, 5.5, 7.5, 8.5, 9.5, 12.0

Bài tập 1

Chương trình kiểm tra một ngày tháng nhập vào có hợp lệ hay không? Người cần nhập vào năm, tháng, ngày của ngày muốn kiểm tra.

Ví dụ 12/12/2017 là ngày hợp lệ, 32/12/2017 là ngày không hợp lệ.

Phân tích các phân vùng tương đương cho yêu cầu trên.

Bài tập 2

Chức năng đăng ký tài khoản của một ứng dụng học tập yêu cầu mật khẩu phải có chứa chữ hoa, chữ thường, số, ký tự đặc biệt và chiều dài tối thiểu là 6.

Sử dụng kỹ thuật phân vùng tương đương viết các test case kiểm tra ô nhập liệu mật khẩu.

Bài tập 3

Để kỷ niệm sinh nhật công ty vào tháng 2, một siêu thị mini chuyên kinh doanh đồ chơi trẻ em sẽ giảm giá 50% cho những hoá đơn mua hàng trong tháng 2 hàng năm có giá trị hoá đơn từ 500k VNĐ trở lên.

Sử dụng phương pháp phân vùng tương đương và phân tích giá trị biên, hãy thiết kế test case kiểm thử khách hàng số tiền (VNĐ) khách hàng được giảm trên hoá đơn?

Kiểm thử phần mềm

PHÂN TÍCH GIÁ TRỊ BIÊN
(Boundary Value Analysis)

Phân tích giá trị biên

Phân tích giá trị biên (BVA) là kỹ thuật kiểm thử dựa **các giá trị tại biên** giữa các phân vùng tương đương, bao gồm trường hợp:

- Hợp lệ (valid).
- Không hợp lệ (invalid).

Phân tích giá trị biên

Một số quy tắc xác định giá trị biên khi thiết kế test case

- Giá trị nhỏ nhất (biên dưới)
- Giá trị nhỏ thứ hai (cận biên dưới)
- Giá trị lớn nhất (biên trên)
- Giá trị lớn thứ hai (cận biên trên)
- Giá trị bình thường

Ví dụ 1: trở lại ví dụ xếp loại học sinh

Ví dụ 1

Để áp dụng phân tích giá trị biên vào kiểm thử chương trình lấy giá trị lớn nhất và nhỏ nhất trong các phân đoạn valid.

- Đoạn "yếu, kém": 0 và 4.9
- Đoạn "trung bình": 5 và 6.9
- Đoạn "khá": 7 và 7.9
- Đoạn "giỏi": 8 và 8.9
- Đoạn "xuất sắc": 9 và 10

Lấy giá trị đầu tiên trong đoạn > 10 là 10.1

Lấy giá trị cuối cùng trong đoạn < 0 là -0.1

Ví dụ 2

Để minh họa lợi ích của giá trị biên xét thêm ví dụ: một developer đã phát triển xong chức năng **kiểm tra số nguyên dương n có phải là số nguyên tố không** và giao cho một tester tiến hành kiểm thử chức năng này với màn hình console cho phép nhập số nguyên dương n và xuất kết quả thông báo số đó có phải nguyên tố hay không.

Ghi chú: số nguyên tố là số tự nhiên chỉ có hai ước số dương phân biệt là 1 và chính nó.

Ví dụ 2

Một **tester ít kinh nghiệm** kiểm thử bằng cách chạy một số giá trị nguyên n để kiểm tra chương trình trên.

C:\WINDOWS\system32\cmd.exe

```
n = 2
NGUYEN TO
n = 7
NGUYEN TO
n = 13
NGUYEN TO
n = 113
NGUYEN TO
n = 71
NGUYEN TO
```

C:\WINDOWS\system32\cmd.exe

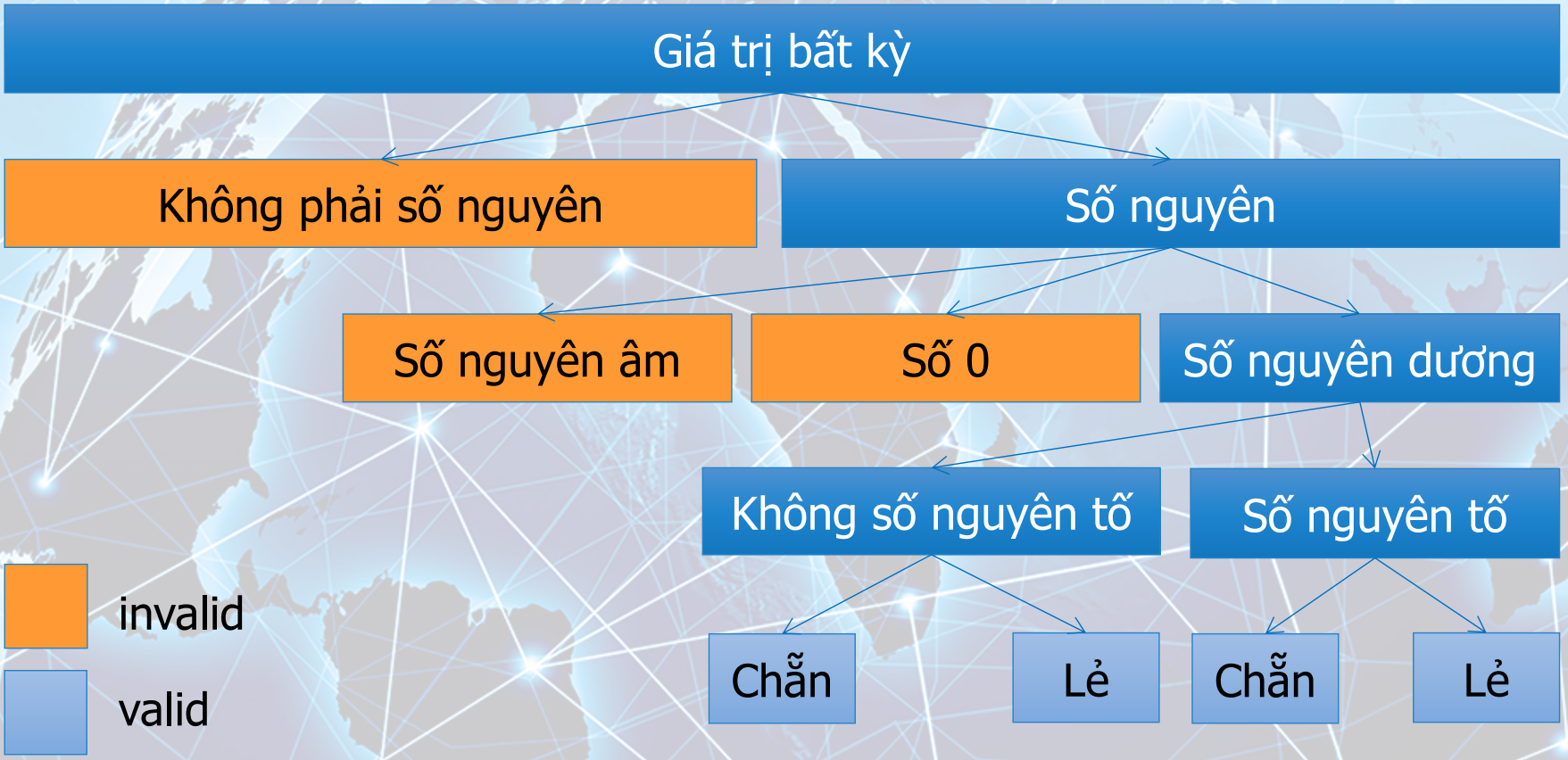
```
n = 12
KHONG NGUYEN TO
n = 143
KHONG NGUYEN TO
n = 112
KHONG NGUYEN TO
n = 58
KHONG NGUYEN TO
n = 98
KHONG NGUYEN TO
```

C:\WINDOWS\system32\cmd.exe

```
n = 0
KHONG NGUYEN TO
n = 1
KHONG NGUYEN TO
n = -7
KHONG NGUYEN TO
n = -6
KHONG NGUYEN TO
n = -53
KHONG NGUYEN TO
```

Ví dụ 2

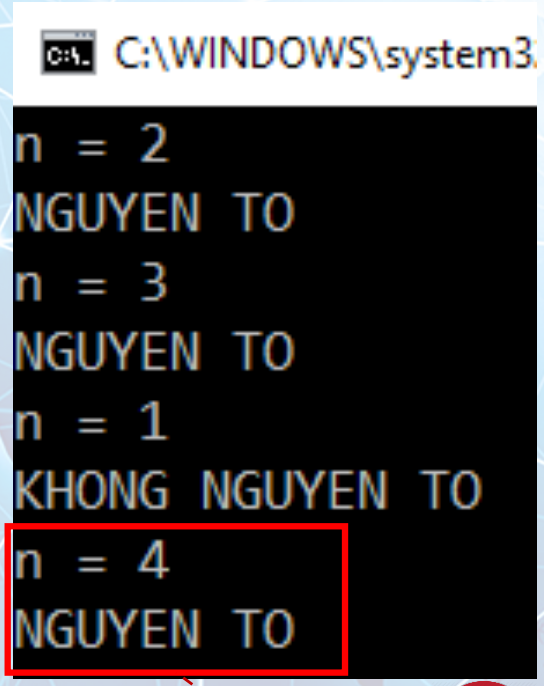
Một tester có kinh nghiệm tiến hành phân vùng tương đương để thiết kế test case kiểm thử chương trình như sau:



Ví dụ 2

Áp dụng phân tích giá trị biên cho phân vùng số nguyên dương, chọn các giá trị biên sau đại diện cho các phân vùng để kiểm thử:

- Phân vùng số nguyên tố chẵn: **2**
- Phân vùng số nguyên tố lẻ: **3**
- Phân vùng số lẻ không phải số nguyên tố: **1**
- Phân vùng số chẵn không là số nguyên tố: **4**



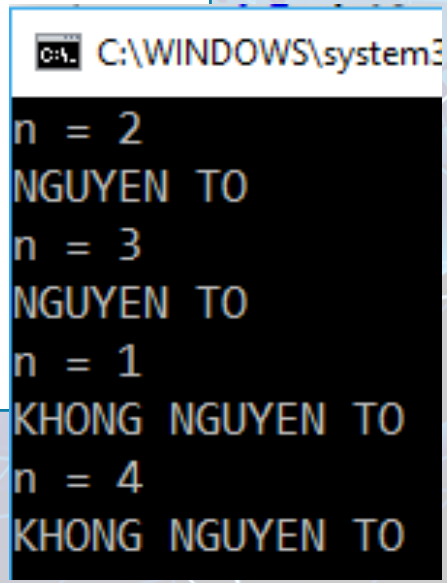
Ví dụ 2

Developer mở chương trình kiểm tra

```
bool ktNguyenTo(int n) {
    if (n < 2)
        return false;

    for (int i = 2; i <=sqrt(n); i++)
        if (n % i == 0)
            return false;

    return true;
}
```



Chương trình minh họa bằng C++

Ưu điểm và khuyết điểm

Ưu điểm

- Đơn giản.
- Hiệu quả cho các hàm có biến độc lập.
- Có thể tự động sinh test case khi xác định được giá trị biên của các biến.

Khuyết điểm

- Không quan tâm đặc trưng của hàm, ngữ nghĩa các biến, cũng như quan hệ giữa các biến.
- Khó áp dụng cho trường hợp các biến có quan hệ ràng buộc nhau.

Bài tập

Một hệ thống ngân hàng trực tuyến của ngân hàng ABC quy định không được chuyển khoản quá 10 triệu (tr) trong ngày, tối thiểu mỗi lần chuyển khoản là 1tr.

Sử dụng phương pháp phân tích giá trị biên thiết kế test case để kiểm tra số tiền chuyển khoản của khách hàng A có được phép chuyển trong ngày hiện tại không?

Giả sử tài khoản người chuyển luôn đủ tiền chuyển.

BẢNG QUYẾT ĐỊNH

(Decision Table)

Bảng quyết định

Bảng quyết định (DT) là kỹ thuật kiểm thử hộp đen dùng xác định **những kịch bản** (scenario) kiểm thử cho những trường hợp có logic nghiệp vụ phức tạp.

DT giúp tester xác định hiệu quả sự **kết hợp các đầu vào khác nhau** với các tình trạng phần mềm thực thi đúng quy tắc nghiệp vụ.

DT nên được sử dụng trong những chương trình có **nhiều lệnh rẽ nhánh** và **các biến đầu vào có mối quan hệ với nhau**.

Cấu trúc bảng quyết định

Các điện kiện: một biến, một quan hệ hoặc một mệnh đề.

Các giá trị điều kiện: một giá trị nào đó của điều kiện (các cột quy tắc - rule).

Các điều kiện	Các giá trị điều kiện
Hành động	Xảy ra hay không

Hành động: một thủ tục hoặc thao tác cần thực hiện.

Các giá trị của hành động: một hành động có thể xảy ra phụ thuộc vào tổ hợp các giá trị điều kiện.

Xây dựng bảng quyết định

Liệt kê tất cả **các điều kiện/đầu vào**.

Tính **số lượng kết hợp** các giá trị của các điều kiện/ đầu vào và đặt các kết hợp đó vào trong phần giá trị các điều kiện.

Xác định các test case tương ứng cho các điều kiện được thỏa mãn.

Các hành động chính là kết quả mong đợi của test case.

Chú ý: thứ tự các điều kiện và thứ tự thực hiện hành động là không quan trọng.

Xây dựng bảng quyết định

Ví dụ: xây dựng bảng quyết định cho chức năng login với hai thông tin đầu vào là username và password trên một ứng dụng web.

Xác định các điều kiện/đầu vào: username và password.

Mỗi đầu vào trên nhận một trong 3 giá trị: rỗng (blank - B), hợp lệ (valid - V) và không hợp lệ (invalid - I).

Số kết hợp giá trị các điều kiện có thể xảy ra là 9

Xây dựng bảng quyết định

Các điều kiện	Username	B	B	B	I	I	I	V	V	V
	Password	B	I	V	B	I	V	B	I	V
Các hành động	Thông điệp lỗi	M1	M1	M1	M3	M3	M3	M2	M4	
	Chuyển đến trang	L	L	L	L	L	L	L	L	H

M1: Vui lòng nhập username

M2: Vui lòng nhập password

M3: Username không hợp lệ

M4: Password không hợp lệ

L: Trang login

H: Trang home

Rút gọn bảng quyết định

Rút gọn các kết hợp các giá trị đầu vào.

Những trường hợp (cột quy tắc) có **cùng giá trị hành động**, nhưng chỉ **khác giá trị của một điều kiện** duy nhất.

Chuyển giá trị của điều kiện khác nhau đó thành "-" và gôm các cột lại thành một.

Lặp lại hai bước trên cho đến khi không còn các cột nào như thế.

Rút gọn bảng quyết định

Các điều kiện	Username	B	B	B	I	I	I	V	V	V
	Password	B	I	V	B	I	V	B	I	V
Các hành động	Thông điệp lỗi	M1	M1	M1	M3	M3	M3	M2	M4	
	Chuyển đến trang	L	L	L	L	L	L	L	L	H

Các điều kiện	Username	B	I	V	V	V
	Password	-	-	B	I	V
Các hành động	Thông điệp lỗi	M1	M3	M2	M4	
	Chuyển đến trang	L	L	L	L	H

Chuyển thành các test case

Các quy tắc chuyển từ bảng quyết định thành các test case:

Nếu giá trị các điều kiện nhập là các giá trị **luận lý** (true/false) thì **mỗi cột** quy tắc được chuyển thành **một test case**.

Nếu giá trị các điều kiện nhập có **nhiều giá trị** thì **mỗi cột** quy tắc được chuyển thành **nhiều test case** sử dụng kỹ thuật phân vùng tương đương hoặc phân tích giá trị biên.

Bài tập 1

Cho chương trình xác định một tam giác có phải là tam giác cân không, biết người dùng nhập vào chiều dài 3 cạnh a, b, c của tam giác.

Sử dụng bảng quyết định thiết kế test case để kiểm thử chương trình trên.

Bài tập 2

Một thư viện ABC có chức năng cho phép độc giả mượn sách. Theo đó, độc giả mượn sách không được quá 500 quyển sách trong năm (không phân biệt tên đầu sách), nhưng không được phép mượn quá 5 quyển trong một lần mượn, và phải trả các cuốn sách đã mượn mới được phép mượn tiếp nữa.

Sử dụng bảng quyết định và phân tích giá trị biên thiết kế test case kiểm thử độc giả có được phép mượn sách không và được mượn tối đa bao nhiêu quyển trong lần mượn mới?

Bài tập 3

Một hệ thống quản lý học tập có chức năng xét học bổng cho sinh viên vào cuối mỗi học kỳ, biết sinh viên có học bổng nếu không rớt môn nào trong học kỳ đó và có điểm trung bình học kỳ từ 7.0 trở lên. Điểm tính trên thang điểm 10, điểm qua môn tính từ 5 trở lên.

Ngoài ra, đối với các môn học có điểm ≥ 4 và < 5 cũng sẽ được xem là qua môn nếu điểm trung bình học kỳ đó là từ 7.5 trở lên.

Sử dụng bảng quyết định và phân tích giá trị biên thiết kế test case kiểm thử một sinh viên có được nhận học bổng trong học kỳ hay không, biết đầu vào là điểm các môn học trong học kỳ đang xét (các điểm trung bình được làm tròn một chữ số thập phân)?

Bài tập 4

Một chương trình khuyến mãi tri ân khách hàng của hãng A cho những khách hàng mua dòng điện thoại cao cấp của hãng diễn ra từ ngày 20/11/2022 đến hết ngày 31/12/2022 – dòng điện thoại được gọi là cao cấp nếu giá bán lớn hơn hoặc bằng 20tr. Theo đó, nếu khách hàng mua điện thoại cao cấp của hãng A trong khoảng thời gian đó sẽ được tặng 1 loa bluethooth và miếng dán màn hình. Ngoài ra đối với những khách hàng đã từng dùng dòng điện thoại cao cấp của hãng A, tính từ thời điểm đã mua cho đến thời điểm mua mới, nếu khoảng thời gian này

Không quá 1 năm thì khách hàng sẽ được giảm thêm 2 triệu trên giá sản phẩm.

Từ trên 1 năm đến nhỏ hơn hoặc bằng 2 năm thì khách hàng được giảm thêm 1 triệu trên giá sản phẩm.

Bài tập 4

Sử dụng các phương pháp bảng quyết định và phân tích giá trị biên thiết kế các test case kiểm thử các khuyến mãi mà người dùng nhận được khi mua điện thoại cao cấp hãng A?

Kiểm thử phần mềm

CHUYỂN ĐỔI TRẠNG THÁI
(State Transition)

Chuyển đổi trạng thái

Các khía cạnh của hệ thống được mô tả thông qua **máy trạng thái hữu hạn**.

Hệ thống xử lý cùng đầu vào, nhưng cho kết quả đầu ra khác nhau gọi máy trạng thái hữu hạn.

Máy trạng thái hữu hạn được biểu diễn giống như **lược đồ trạng thái**.

Hệ thống sẽ có nhiều trạng thái khác nhau, sự dịch chuyển từ một trạng thái này sang trạng thái khác được quyết định bởi một sự kiện nào đó.

Chuyển đổi trạng thái

Một mô hình chuyển đổi trạng thái có 4 phần cơ bản:

Các trạng thái (**states**) phần mềm có thể xảy ra.

Sự dịch chuyển (**transitions**) từ trạng thái này sang trạng thái khác.

Các sự kiện (**events**) dẫn đến sự dịch chuyển trạng thái.

Các hành động (**actions**) là kết quả của việc dịch chuyển trạng thái.

Ví dụ hệ thống bug tracking

Một quy trình sửa lỗi (fix bug) của một hệ thống bug tracking ở một công ty phần mềm như sau:

Tester phát hiện bug và tạo báo cáo bug bắt đầu với trạng thái "Open".

Developer xem xét nếu thấy nó không phải bug thì chuyển bug sang thái "Wont Fix" và giải thích cho tester.

Nếu tester cũng đồng ý đó không phải bug thì chuyển bug sang trạng thái "Closed", ngược lại chuyển về trạng thái "Open".

Ví dụ hệ thống bug tracking

Nếu developer xem qua thấy bug được tạo đúng là lỗi phần mềm và tiến hành fix bug thì chuyển bug sang trạng thái "In Progress".

Sau khi fix bug xong, developer chuyển nó sang trạng thái "Testing" để tester tiến hành xác nhận (verify) thật sự bug đã được fix.

Trong quá trình đang fix bug, developer nhận ra nó không phải lỗi phần mềm thì developer chuyển về trạng thái "Wont Fix".

Ví dụ hệ thống bug tracking

Nếu tester kiểm tra qua bug đã được fix và thấy ổn thì chuyển bug sang trạng thái "Closed", ngược lại chuyển nó về trạng thái "Open" và yêu cầu developer fix lại.

Sau khi bug đã đóng, nhưng quá trình test sau đó lại thấy nó tái xuất hiện thì tester có thể chuyển nó về trạng thái "Open" và yêu cầu developer tiếp tục fix.

Ví dụ hệ thống bug tracking

Các trạng thái: O = Open, IP = In Progress, WF = Wont Fix, T = Testing, C = Closed.

Các dịch chuyển:

- Open → In Progress, Wont Fix
- In Progress → Wont Fix, Testing
- Wont Fix → Open, Close
- Testing → Open, Closed
- Closed → Open

Ví dụ hệ thống bug tracking

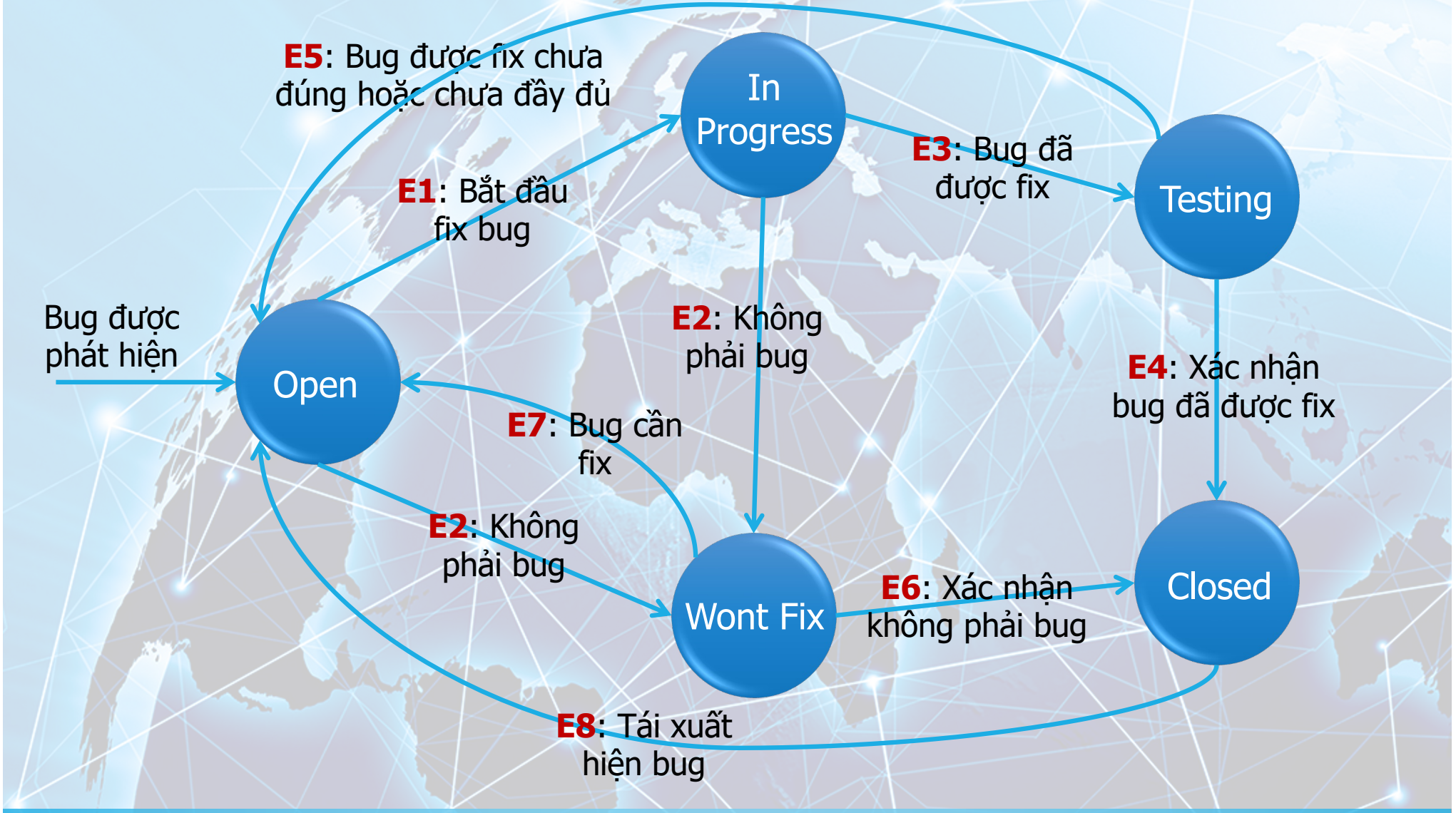
Các sự kiện

- E1: bắt đầu fix bug
- E2: không phải bug
- E3: bug đã được fix
- E4: xác nhận bug đã được fix
- E5: bug được fix chưa đúng hoặc chưa đầy đủ
- E6: xác nhận không phải bug
- E7: bug cần fix
- E8: tái xuất hiện bug

Ví dụ hệ thống bug tracking

Các hành động kết quả: khi có bất cự sự thay đổi trạng thái nào của bug, hệ thống sẽ gửi email thông báo đến tất cả các thành viên có liên quan đến bug đó về trạng thái hiện tại của bug.

Ví dụ hệ thống bug tracking



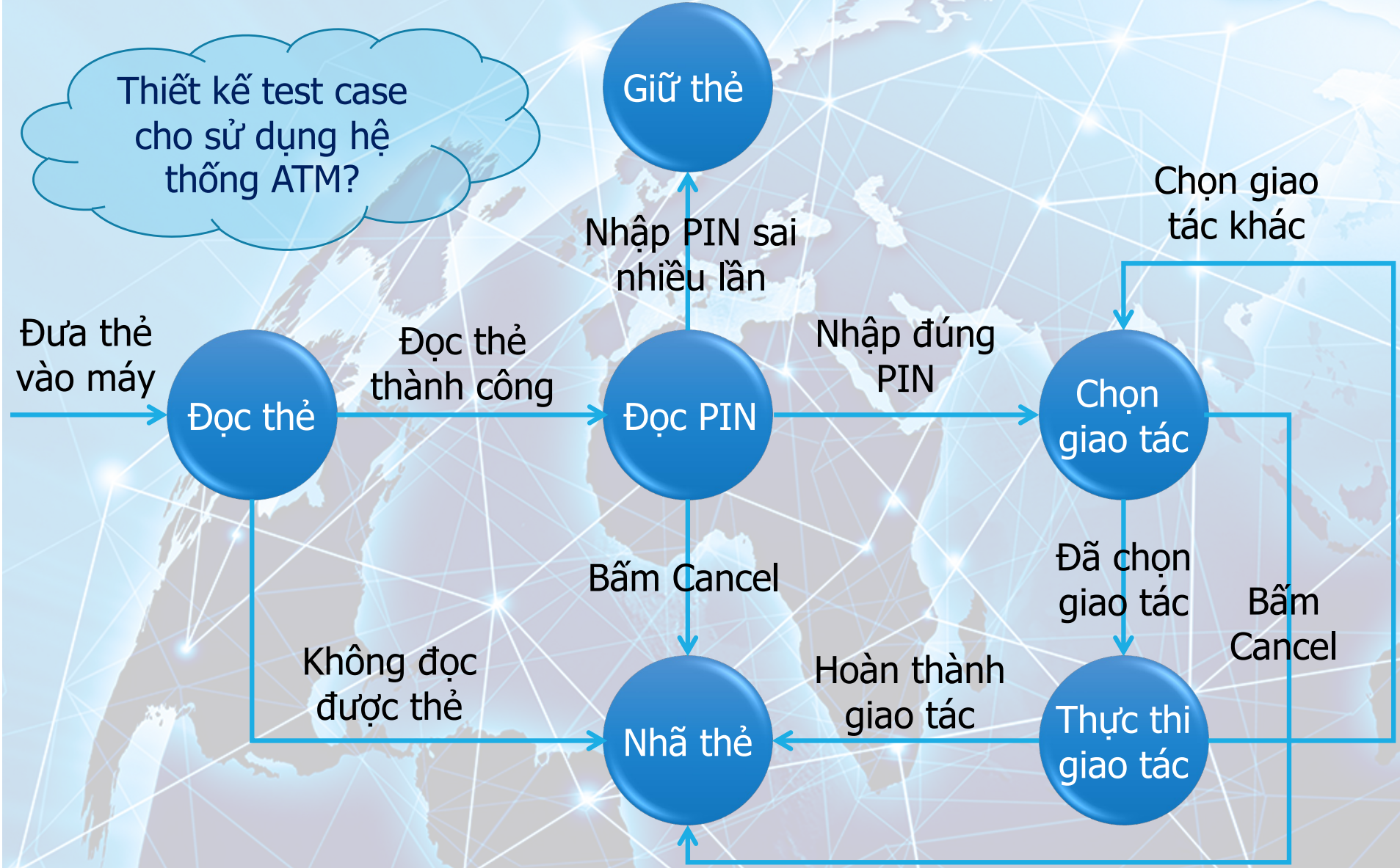
Ví dụ hệ thống bug tracking

Bảng mô tả trạng thái

	E1	E2	E3	E4	E5	E6	E7	E8
O	IP	WF						
IP		WF	T					
T				C	O			
C								O
WF						C	O	

Dựa vào bảng trạng thái có thể thiết kế 9 test case hợp lệ: O → IP, O → WF, IP → WF, IP → T, T → C, T → O; C → O, WF → C, WF → O và một vài trường hợp không hợp lệ.

Ví dụ sử dụng hệ thống ATM



Bài tập 1

Một hệ thống quản lý cho phép gửi và nhận tin nhắn trong hệ thống, khi người dùng nhận tin mới có trạng thái là tin chưa đọc, nếu người nhận mở ra đọc thì tin đó thành trạng thái đã đọc. Sau khi đọc tin, người dùng cũng có thể chuyển nó thành tin chưa đọc để ghi nhớ. Ngoài ra, người dùng cũng có thể xóa tin tức, ban đầu tin xóa tạm nằm trong thùng rác, trong 24h kể từ lúc xóa người dùng có thể phục hồi lại trạng thái trước khi xóa, sau khoảng thời gian này tin sẽ bị xóa vĩnh viễn.

Vẽ lược đồ chuyển đổi trạng thái tin nhắn và viết các test case cho chúng.

Bài tập 2

Một hệ thống quản lý học tập có chức năng cho phép đăng bài viết, một bài viết khi đăng mới chỉ được phép cập nhật hoặc xóa trong vòng 15 phút kể từ lúc submit đăng bài, sau khoảng thời gian này bài viết không được phép chính sửa hay xóa nửa và bài viết sẽ tự động được xuất bản trên hệ thống để người khác có thể đọc. Ngoài ra, khi vừa soạn xong bài viết hoặc trong vòng 15 phút từ lúc submit bài viết, tác giả bài viết có quyền bấm nút "Publish" để xuất bản bài viết, và tất nhiên không được xóa hoặc cập nhật bài viết sau khi đã xuất bản. Sau khi một bài viết được xuất bản, tác giả bài viết muốn xóa hoặc cập nhật bài viết cần phải liên hệ với admin thực hiện. Chú ý sau khi admin chỉnh sửa bài viết đã xuất bản, bài viết đó vẫn ở trạng thái xuất bản để người khác đọc.

Bài tập 3

Notification Center là một module trong hệ thống học tập trực tiếp, một người sẽ nhận được thông báo mới khi người khác bình luận, thích trên bài viết họ đã đăng hoặc người khác cũng bình luận trên bài viết mà họ đã bình luận không quá 1 tháng. Ngoài ra nếu người dùng đã theo dõi (follow) một bài viết thì luôn nhận thông báo khi có người bình luận hoặc thích bài viết đó. Một thông báo mới có trạng thái là UNSEEN, khi người dùng mở Notification sẽ thấy những thông báo này có màu nền xanh và chữ đậm, và những thông báo này sẽ chuyển thành trạng thái SEEN và có màu nền xanh mờ hơn so với UNSEEN và chữ không in đậm. Nếu người dùng click trên thông báo đó để mở bài viết ra xem thì thông báo đó trở thành trạng thái OPENED, hiển thị không có màu nền và màu chữ bình thường.

Vẽ lược đồ dịch chuyển đổi trạng thái các thông báo trong Notification Center và thiết kế các test case kiểm thử.

Kiểm thử phần mềm

KIỂM THỬ DỰA TRÊN LƯỢC ĐỒ USE CASE

Kiểm thử dựa trên use case

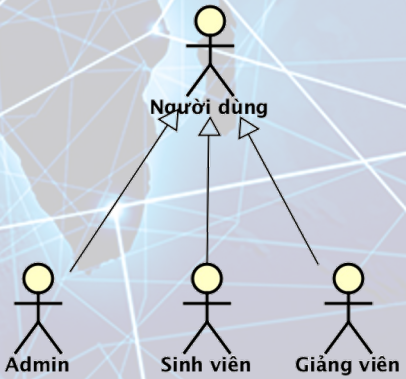
Thiết kế test case dựa trên use case sẽ hữu dụng trong việc thực hiện các **quy tắc nghiệp vụ hoặc các luồng xử lý**, phát hiện được những lỗ hổng, điểm yếu của hệ thống khi kiểm thử từng thành phần riêng rẽ không phát hiện ra.

Kỹ thuật này là nền tảng phát triển các test case trong cấp độ **kiểm thử hệ thống** và **kiểm thử chấp nhận** do các use case đại diện cho khả năng **sử dụng thực sự** của người dùng.

Lược đồ use case

Lược đồ use case nhằm mô tả các chức năng gì của hệ thống được thực hiện bởi actor nào và vai trò của actor trong hệ thống.

Giữa hai actor có thể có mối quan hệ tổng quát hoá và chuyên biệt hoá.



Lược đồ use case

Quan hệ giữa các use case

Use Case S luôn được thực hiện khi Use case Base được gọi

Use Case S thỉnh thoảng được thực hiện khi Use case Base được gọi

Use case Child là trường hợp chuyên biệt, nâng cấp của use case Base

Đặc tả use case

Thông thường lược đồ use case sẽ có các đặc tả use case làm cơ sở hữu ích cho kiểm thử viên thiết kế test case.

Use case testing cần kiểm thử kịch bản thành công của use case và cho các kịch bản thay thế, ngoại lệ của use case.

Một đặc tả use case thường có các thông tin sau:

- Use case id để định danh use case.
- Tên use case.
- Mô tả vắn tắt mục đích use case.

Đặc tả use case

Các actor chính.

Các actor phụ.

Tiền điều kiện (Pre-conditions).

Hậu điều kiện (Post-conditions).

Luồng hoạt động chính (Main Flows).

Luồng hoạt động thay thế (Alternative flows)

Luồng hoạt động ngoại lệ (Exception flows)

KIỂM THỬ DỰA TRÊN KINH NGHIỆM

Đoán lỗi

Đoán lỗi (error guessing) là một kỹ thuật đơn giản và thường được áp dụng **sau khi tất cả** các kỹ thuật chính thức đã được áp dụng.

Kỹ thuật này dựa trên **kỹ năng, kinh nghiệm, trực giác** của kiểm thử viên đã kiểm thử các ứng dụng tương tự nhằm xác định những test case đặc biệt không được phát hiện khi áp dụng các kỹ thuật chính thức trước đó.

Những test case tập trung những điểm yếu đáng nghi của hệ thống hoặc những khía cạnh của hệ thống đã từng tìm thấy vấn đề khi kiểm thử các hệ thống tương tự trước đây.

Kiểm thử thăm dò

Kiểm thử thăm dò (exploration testing) làm rõ hệ thống có gì, không có gì, cái nào làm việc tốt, cái nào làm việc chưa tốt để kiểm thử viên có cơ sở đưa ra quyết định những gì cần kiểm thử tiếp theo.

Đây là kỹ thuật tiếp cận có cấu trúc kết hợp kiến thức, sự quan sát, kinh nghiệm của các kiểm thử viên để kiểm thử những chỗ đặc tả yêu cầu bị **thiếu, không đầy đủ** (poor specifications) hoặc có nhiều áp lực thời gian thực hiện.

Kiểm thử thăm dò

Kỹ thuật này **tập trung thực thi** kiểm thử hơn là việc lên kế hoạch kiểm thử. Việc thiết kế test case và thực thi test case được thực hiện song song, và cũng **không cần một tài liệu chính quy** về test condition, test case hoặc test script.

Kiểm thử thăm dò được thực hiện trong các **giai đoạn đầu** của vòng đời phát triển phần mềm khi mã nguồn trải qua những **thay đổi nhanh chóng**, kiểm thử thăm dò sẽ phát huy hiệu quả.

Lập trình viên sử dụng kiểm thử thăm dò để thực hiện các Unit Test, còn **kiểm thử viên** có thể làm quen hệ thống bằng kỹ thuật này.

Câu hỏi

Đoán lỗi (error guessing) được sử dụng tốt nhất khi:

A. Sau khi hệ thống được triển khai.

B. Sau khi các kỹ thuật kiểm thử chính thức đã được áp dụng.

C. Thực hiện bởi những tester ít kinh nghiệm.

D. Chỉ được thực hiện bởi người dùng cuối.

Q&A