

Lecture 4: Black Box Approach for Test Case Designing

Kỹ thuật kiểm thử hộp đen

IT4501

Bùi Thị Mai Anh (anhbtm@soict.hust.edu.vn)

Bộ môn Công nghệ phần mềm

Viện công nghệ thông tin và truyền thông

1

Contents

Những nội dung chính

2

- Equivalence class partitioning
- Boundary value analysis
- Decision Table
- State Transition Testing

2

What is Black box testing

3

- Kiểm thử hộp đen là một chiến lược trong đó việc kiểm thử được thực hiện dựa trên yêu cầu và đặc tả của chức năng
- Xác minh đầu ra của chức năng mà không quan tâm tới cấu trúc bên trong.
- Chúng ta sẽ không biết lỗi được
 - Việc kiểm soát bên trong như thế nào và
 - Đầu vào dữ liệu đưa vào để kiểm chứng cũng không thể đảm bảo kiểm thử được tất cả các đường dẫn của code như các chiến lược kiểm thử hộp trắng

Black Box vs. White Box

4

- **Black Box: Kiểm thử chức năng**
 - Unit test
 - Integration test
 - System test
 - Programmers & Test Engineers & Quality Assurance Engineers
 - **White Box: Kiểm thử cấu trúc**
 - Unit test
 - Integration test
 - Programmers & Test Engineers
- | | | |
|-----|---|---|
| USE | <ul style="list-style-type: none"> • External/user view: <ul style="list-style-type: none"> – Check conformance with specification • Abstraction from details: <ul style="list-style-type: none"> – Source code not needed • Scales up: <ul style="list-style-type: none"> – Different techniques at different levels of granularity | <ul style="list-style-type: none"> • Internal/developer view: <ul style="list-style-type: none"> – Allows tester to be confident about test coverage • Based on control or data flow: <ul style="list-style-type: none"> – Easier debugging • Does not scale up: <ul style="list-style-type: none"> – Mostly applicable at unit and integration testing levels |
| | BOTH! | |

Equivalence Class Partitioning Phân chia lớp tương đương

5

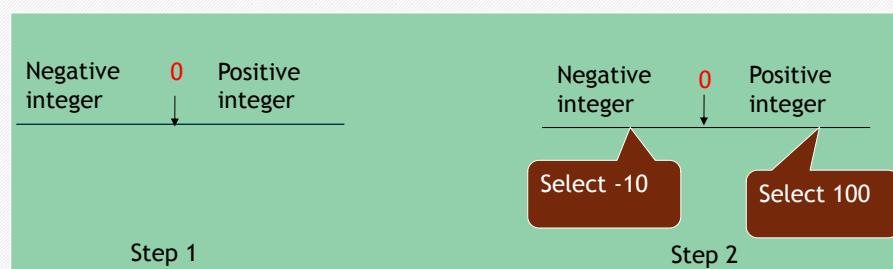
- Phương pháp chia mien dữ lieu đầu vào của chương trình thành các lớp tương đương
- Một lớp tương đương được định nghĩa là:
 - Tập hợp các điểm dữ liệu cho phép chương trình có cùng 1 hành xử (behavior)
 - Tất cả các điểm dữ liệu trong lớp là tương đương nhau
- Hai lớp tương đương cơ bản:
 - Valid class
 - Invalid class

5

Equivalent Class Testing Kiểm thử lớp tương đương

6

- **Step 1:** Xác định tất cả các phân lớp tương đương từ yêu cầu đặc tả của chương trình
- **Step 2:** Với mỗi lớp tương đương, lấy ra 1 điểm dữ liệu đặc trưng để xây dựng test case



6

Example: Insurance System

7

- **Đặc tả:**
 - Hệ thống sẽ từ chối hồ sơ bảo hiểm của những người lớn tuổi
 - Tuổi chấp nhận bảo hiểm của nam giới là ≤ 80
 - Tuổi chấp nhận bảo hiểm của nữ giới là ≤ 85
 - Tuổi của mỗi người nằm trong miền từ 0-99
- Có bao nhiêu phân lớp tương đương

7

Answer

8

- Input: Gender & Age
- Output: accept/reject

Classes	Test Cases
C1: Input: Males over 80	T1: male, 83, reject
C2: Input: Males 80 or under	T2: male, 56, accept
C3: Input: Females 85 or under	T3: female, 83, accept
C4: Input: Females over 85	T4: female, 87, reject

- What's about the invalid data?

8

Equivalence Class Testing Guidelines

Một số chú ý khi sử dụng phân lớp tương đương

9

Nếu điều kiện của input là:

- 1 miền dữ liệu, e.g., $x = [0..9]$
- 1 danh sách các giá trị sắp xếp, e.g., $\text{owner} = \{1, 2, 3, 4\}$
- ==> Định nghĩa được 1 lớp valid và 2 lớp invalid
 - 1 tập hợp các dữ liệu cho trước, e.g., $\text{vehicle} = \{\text{car}, \text{motorcycle}, \text{truck}\}$
 - Thoả mãn một điều kiện boolean, e.g., "first character of the identifier must be a letter"
- ==> Định nghĩa được 1 lớp valid và 1 lớp invalid
 - Là những loại khác
- ==> Phân chia cụ thể hơn

9

Exercise 1: Program to determine employability

10

Age	Employment Status
0-15	Don't hire
16-17	Can hire part-time
18-55	Can hire full-time
56-99	Don't hire

How many equivalent classes? How many test case should be designed?

10

Exercise 2: Examination Judgment Program

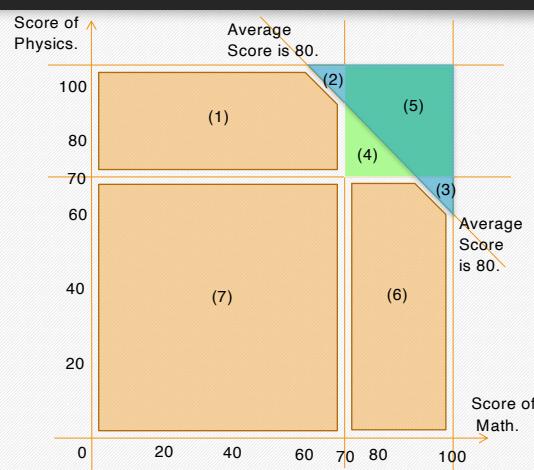
11

- Đặc tả
 - 2 môn học: Maths và Physics
 - Sinh viên sẽ đỗ nếu:
 - Điểm của cả hai môn đều trong dải giá trị từ 70 đến 100
 - Hoặc trung bình điểm của hai môn nằm trong dải từ 80 đến 100
 - Trượt nếu không thỏa mãn hai điều kiện trên

11

Exercise 2

12



Score	Math.	Physics	Result
(1)	55	85	Failed
(2)	67	97	Passed
(3)	96	68	Passed
(4)	77	80	Passed
(5)	85	92	Passed
(6)	79	58	Failed
(7)	52	58	Failed

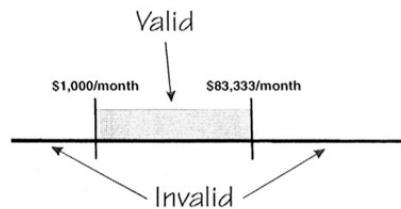
12

Advantage of Equivalence Partitioning

13



Reduce the number of test case



Ensure each class is tested

13

Disadvantages of Equivalence Partitioning

14

- Việc xây dựng các lớp tương đương phụ thuộc vào kinh nghiệm của kỹ sư kiểm thử
- Không xem xét các giá trị tại biên

14

Boundary Value Analysis Phân tích giá trị biên

15

- Phương pháp này phụ trợ cho phân chia lớp tương đương khi thiết kế các ca kiểm thử
- Cho phép lựa chọn các ca kiểm thử để kiểm tra tại giá trị biên của các lớp tương đương
- Steps:
 - Xác định các phân lớp tương đương
 - Xác định giá trị biên của các lớp tương đương
 - Tạo các ca kiểm thử cho mỗi một giá trị tại biên, một giá trị bé hơn biên và một giá trị lớn hơn biên.

15

Example

- Employability Module
 - 6 equivalence classes (4 valid classes)
 - For each boundary value, pick up 3 values on/below/upper
- 16 Test cases

Age	Employment status
0-15	Don't hire
16-17	Can hire part-time
18-54	Can hire full-time
55-99	Don't hire



16

Example: Human's age

17

- Một cách khác để lựa chọn giá trị biên
 - Một giá trị trên biên
 - Một giá trị sát biên thuộc phân lớp invalid



17

Advantage of Boundary Value Analysis Ưu điểm

18

- Dễ thực hiện
- Giúp hoàn thiện phân lớp tương đương
- Đảm bảo không bỏ sót điều kiện khi xây dựng các ca kiểm thử

18

Limitations Hạn chế

19

- Chỉ thích hợp khi dữ liệu là một miền (range) hoặc một tập hợp (set)
- Khi quan hệ giữa các input là phức tạp thì không thể thực hiện được

19

Example: The nextDate function

20

- Problem statement:
 - nextDate nhận 3 tham số day/month/year của ngày hiện tại và trả về ngày tiếp theo:
 - C₁: 1 ≤ M ≤ 12
 - C₂: 1 ≤ D ≤ 31
 - C₃: 1850 ≤ Y ≤ 2050
- How many equivalent classes there are?

20

Equivalent Classes

21

Valid ECs	
M1 = {month: 1 <= month <= 12}	M2 = {month: month < 1}
D1={day:1<=day<=31}	M2 = {month: month > 12}
Y1 = {year: 1850 <= year <= 2050}	D2={day:day<1}

Can we define better equivalence classes?

21

Equivalent Classes

22

- Which months have 30 days? 31 days?
- February has 28 or 29 days depend on whether year is leap year or not

Valid ECs
M1 = {month has 30 days}
M2 = {month has 31 days}
M3 ={February}
D1={1<=day<=27}, D2={day=28}
D3= {day = 29}, D4= {day = 30}, D5 {day = 31}
Y1 = {year: year is a non-leap year}
Y2 = {year: year is a leap year}

22

Design Test cases

23

Case ID	Month	Day	Year	Expected Output
C1	-1	15	1902	Value of month not in range
C2				
C3				
C4				
...				

There are too many combinations of M, D and Y ➔ a lot of test case

23

Decision Table Technique
Kĩ thuật sử dụng bảng quyết định

24

24

Decision Table Bảng quyết định

25

- Là một công cụ xuất sắc để mô tả các yêu cầu có quan hệ chặt chẽ với nhau của hệ thống
- Là 1 công cụ để xây dựng các ca kiểm thử

25

General Form

26

		Rule 1	Rule 2	...	Rule p
Conditions	Condition 1				
	Condition 2				
	...				
	Condition n				
Actions	Action 1				
	Action 2				
	...				
	Action m				

26

Example: Car Insurance Discount

27

- Specification
 - If married and number of employed years ≥ 3 then discount 70%
 - If married and number of employed years < 3 then
 - If good student, discount 50%
 - Otherwise, 20%
 - If not married and number of employed years ≥ 3 , discount 60%
 - In case of not married and number of employed years < 3
 - If good student, discount 40%
 - Otherwise 0%

27

Answer

28

		Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
C	Married?	Yes	Yes	Yes	Yes	No	No	No	No
	# years employed	≥ 3	≥ 3	< 3	< 3	≥ 3	≥ 3	< 3	< 3
	Good student?	Yes	No	Yes	No	Yes	No	yes	No
A	Discount	70	70	50	20	60	60	40	0

8 test cases

28

Answer

29

		<u>Rule 1-2</u>	Rule 3	Rule 4	<u>Rule 5-6</u>	Rule 7	Rule 8
C	Married?	Yes	Yes	Yes	No	No	No
	# years employed	≥ 3	<3	<3	≥ 3	<3	<3
	Good student?	-	Yes	No	-	Yes	No
A	Discount	70	50	20	60	40	0

6 test cases

29

Some terms
Một số thuật ngữ

30

		Rule 1-2	Rule 3	Rule 4	Rule 5-6	Rule 7	Rule 8
C	Married?	Yes	Yes	Yes	No	No	No
	# years employed	≥ 3	<3	<3	≥ 3	<3	<3
	Good student?	-	Yes	No	-	Yes	No
A	Discount	70	50	20	60	40	0

Limited entry table – conditions are binary only

Extended entry table – conditions may take on multiple values

Don't-care-entry – condition is irrelevant or not applicable; any value is OK

Decision tables are **declarative** – order between entries does not matter

30

Using Table Decision for Designing Test Cases

Sử dụng bảng quyết định để thiết kế ca kiểm thử

31

- Rules sẽ trở thành các TC
- Conditions cho phép xác định các dữ liệu cụ thể của TC
- Actions trở thành kết quả mong đợi

		Test Case 1	Test Case 2	...	Test Case p
Inputs	Condition 1				
	Condition 2				
	...				
	Condition n				
Expected Results	Action 1				
	Action 2				
	...				
	Action m				

31

Exercise 1: Triangle Program

32

- The program accepts three integers a,b,c as inputs
- These integers are interpreted as representing the lengths of sides of a triangle
- These variables a,b,c must satisfy the following conditions
 - $a < b + c$, $b < a + c$, $c < a + b$
- The program must determine whether the inputs are not a triangle, an isosceles or an equilateral triangle

32

Decision table

33

		Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8	Rule 9
C	a,b,c forms a triangle									
	a=b									
	a=c									
	b=c									
A	NaT									
	Isosceles									
	Scalene									
	Equilateral									

33

Decision Table

34

		Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8	Rule 9
C	a,b,c forms a triangle	N	Y	Y	Y	Y	Y	Y	Y	Y
	a=b	—	Y	Y	Y	Y	N	N	N	N
	a=c	—	Y	Y	N	N	Y	Y	N	N
	b=c	—	Y	N	Y	N	Y	N	Y	N
A	NaT									
	Isosceles									
	Scalene									
	Equilateral									

34

Decision Table

35

		Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8	Rule 9
C	a,b,c forms a triangle	N	Y	Y	Y	Y	Y	Y	Y	Y
	a=b	-	Y	Y	Y	Y	N	N	N	N
	a=c	-	Y	Y	N	N	Y	Y	N	N
	b=c	-	Y	N	Y	N	Y	N	Y	N
A	NaT	X			Impossible	Impossible				
	Isosceles					X		X	X	
	Scalene									X
	Equilateral	X								

35

Design Test Case

36

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c															
	b<c+a															
	c<b+a															
	a=b															
	a=c															
	b=c															
A	NaT															
	Isosceles															
	Scalene															
	Equilateral															

36

Design Test Case

37

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N														
	b<c+a	-														
	a<b+a	-														
	a=b	-														
	a=c	-														
	b=c	-														
A	NaT	X														
	Isosceles															
	Scalene															
	Equilateral															

37

Design Test Case

38

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N														
	b<c+a	-														
	c<b+a	-														
	a=b	-														
	a=c	-														
	b=c	-														
A	NaT	X														
	Isosceles															
	Scalene															
	Equilateral															

38

Design Test Case

39

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N	Y	Y	T								4	2	1	NaT
	b<c+a	-	N	Y	T											
	c<b+a	-	-	N	T											
	a=b	-	-	-	T											
	a=c	-	-	-	T											
	b=c	-	-	-	T											
A	NaT	X														
	Isosceles															
	Scalene															
	Equilateral															

39

Design Test Case

40

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N	Y	Y	T								4	2	1	NaT
	b<c+a	-	N	Y	T								2	4	1	NaT
	c<b+a	-	-	N	T								1	4	2	NaT
	a=b	-	-	-	T								3	3	3	Eq
	a=c	-	-	-	T											
	b=c	-	-	-	T											
A	NaT	X	X	X												
	Isosceles															
	Scalene															
	Equilateral				X											

40

Design Test Case

41

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N	Y	Y	T	T	T						4	2	1	NaT
	b<c+a	-	N	Y	T	T	T						2	4	1	NaT
	c<b+a	-	-	N	T	T	T						1	4	2	NaT
	a=b	-	-	-	T	T	T						3	3	3	Eq
	a=c	-	-	-	T	F	T									
	b=c	-	-	-	T	T	F									
A	NaT	X	X	X												
	Isosceles															
	Scalene															
	Equilateral				X											

41

Design Test Case

42

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N	Y	Y	T	T	T						4	2	1	NaT
	b<c+a	-	N	Y	T	T	T						2	4	1	NaT
	c<b+a	-	-	N	T	T	T						1	4	2	NaT
	a=b	-	-	-	T	T	T						3	3	3	Eq
	a=c	-	-	-	T	F	T									
	b=c	-	-	-	T	T	F									
A	NaT	X	X	X												
	Isosceles															
	Scalene															
	Equilateral				X											

42

Design Test Case

43

		1	2	3	4	5	6	7	8	9	10	11	a	b	c	Exp
C	a<b+c	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	4	2	1	NaT
	b<c+a	-	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	2	4	1	NaT
	c<b+a	-	-	N	Y	Y	Y	Y	Y	Y	Y	Y	1	4	2	NaT
	a=b	-	-	-	Y	Y	Y	Y	N	N	N	N	3	3	3	Eq
	a=c	-	-	-	Y	N	Y	N	Y	Y	N	N				
	b=c	-	-	-	Y	Y	N	N	Y	N	Y	N				
A	NaT	X	X	X									5	5	3	Iso
	Isosceles							X		X	X					
	Scalene											X	5	3	5	Iso
	Equilateral			X									3	5	5	Iso

43

Howmany Test Case for structure testing?

44

```

public static TClass triangleType(int a, int b, int c) {
    if (a <= 0 || b <= 0 || c <= 0) /* p1 */
        return INVALID;
    if (a >= b + c || b >= a+c || c >= a+b) /* p2 */
        return INVALID;
    int count = 0;
    if (a == b) /* p3 */
        count++;
    if (a == c) /* p4 */
        count++;
    if (b == c) /* p5 */
        count++;
    if (count == 0) /* p6 */
        return SCALENE;
    if (count == 1) /* p7 */
        return ISOSCELES;
    return EQUILATERAL;
}

```

○ Identify:

- ① the reachability predicates;
- ② TR(CC) and TR(PC)
- ③ test cases that satisfy a) PC b) CC
- ④ determination predicates for the clauses of p1 and p2
- ⑤ TR(CACC)
- ⑥ test cases that satisfy a) CACC b) RACC [are there infeasible requirements?]

44

Exercise 2: the nextDate function

45

Valid ECs

M1 = {month has 30 days}

M2 = {month has 31 days}

M3 = {February}

D1={1<=day<=27}, D2={day=28}

D3= {day = 29}, D4= {day = 30}, D5 {day = 31}

Y1 = {year: year is a non-leap year}

Y2 = {year: year is a leap year}

45

Decision Table for the nextDate function

46

		1	2	3	4				
C	month in M30								
	month in M31								
	February								
	December								
	1<=day<28								
	Day=28								
	Day=29								
	Day=30								
	Day=31								
	Leap year								
A	Increment Year								
	Increment Month								
	Increment Day								
	Reset year								
	Reset month								
	Reset day								

46

Decision Table for the nextDate function

47

		1	2	3	4			
C	month in M30	T	-	-	-			
	month in M31	-	T	-	-			
	February	-	-	T	-			
	December	-	-	-	T			
	1<=day<28	T	T	-	T			
	Day=28	-	-	T	-			
	Day=29	-	-	-	-			
	Day=30	-	-	-	-			
	Day=31	-	-	-	-			
	Leap year	F	F	F	F			
A	Increment Year							
	Increment Month			X				
	Increment Day	X	X		X			
	Reset year							
	Reset month							
	Reset day			X				

Can we simplify the conditions?

47

Decision Table for the nextDate function

48

		1	2	3	4				
C	month								
	day								
	year								
A	Increment Year								
	Increment Month								
	Increment Day								
	Reset year								
	Reset month								
	Reset day								
	Error								

M31 = { 1,3,5,7,8,10 }; M30 = { 4,6,9,11 }; M2 = { 2 }; M12 = { 12 }; D27={1,2,...,27}; D28={28},D29={29 },D30={30},D31={31} YN = *not a leap year*; YL = *leap year*

48

Decision Table for the nextDate function

49

		1	2	3	4	5	6	7	8	9			
C	month	M30	M30	M30	M31	M31	M12	M12	M2	M2	M2	M2	M2
	day	D27 D28 D29	D30	D31	D27 D28 D29 D30	D31	D27	D31	D30	D27	D28	D28	D29
	year	-	-	-	-	-	-	-	-	YL	YN	YL	YN
A	Increment Year						X						
	Increment Month		X			X					X	X	
	Increment Day	X		X		X			X	X			
	Reset year												
	Reset month		X			X	X						
	Reset day						X			X	X		
	Error			X				X					X

49

Design Test Cases

50

Case ID	Month	Day	Year	Expected output
1-3	April	15	2001	April 16, 2001
4	April	30	2001	May 1, 2001
5	April	31	2001	Error
6-9	January	15	2001	January 16, 2001
10	January	31	2001	February 1, 2001
11-14	December	15	2001	December 16, 2001
15	December	31	2001	January 1, 2002
16	February	15	2001	February 16, 2001
17	February	28	2004	February 29, 2004
18	February	28	2001	March 1, 2001
19	February	29	2004	March 1, 2004
20	February	29	2001	Error
21, 22	February	30	2001	Error

50

Advantages

51

- Nhìn chung bảng quyết định là một công cụ khá mạnh, phù hợp sử dụng với những ứng dụng trong đó có nhiều trường hợp khác nhau của dữ liệu (if then else hoặc switch case)
 - Giữa các biến dữ liệu có các quan hệ logic hoặc quan hệ nhân quả với nhau

IT4501 - Software Engineering Department - SoICT/HUST

11/28/22

51

Disadvantages

52

- Bảng quyết định sẽ gặp khó khăn khi dữ liệu có quá nhiều điều kiện ràng buộc với nhau khiến bảng trở lên rất lớn và cần chia nhỏ thành các bảng bé hơn để loại trừ các trường hợp dư thừa dữ liệu

IT4501 - Software Engineering Department - SoICT/HUST

11/28/22

52

Exercise 3: Road charge

53

- A program that calculates road user charges for foreign heavy vehicles
- Inputs:
 - distance that the driver are intending to drive on a road
 - the emission category of the vehicle (0 or 1 or 2)
 - the number of axles of the truck
- Driver can pay for day or week
- Chargeable period for day is 00.00-24.00. For example, if the journey begins at 21h30 on one day and finishes at 02.20 next day, payment for two days is required

53

Exercise 3: Tarif Tables

54

Toll Tariff 2014

Max axles	3	3	3	4	4	4
Emission Class	0	1	2 or cleaner	0	1	2 or cleaner
1 day	67	67	67	67	67	67
1 week	220	194	169	347	313	279

54

Exercise 4: Bus tarif

55

- A program allows to calculate automatically the monthly bus tarif
- Inputs: age and distance
- Age
 - 0-3 years old: infant charge
 - 4-14 years old: child charge
 - 15-59 years old: adult charge
 - 60-99 years old: silver charge
 - Do not treat at the age of 100 or more
- Distance: <=10 or >10 km

	Distance <=10 km	Distance > 10 km
Infant charge	0	0
Child charge	100	130
Adult charge	200	250
Silver charge	160	200

55

Exercise 4: Bus tarif (in EUR)

56

	Distance <=10 km	Distance > 10 km
Infant charge	0	0
Child charge	100	130
Adult charge	200	250
Silver charge	160	200

56

Exercise 4:

57

- With input distance, list test cases which using Equivalent and Boundary (if possible)
- With input age, list test cases which using Equivalent and Boundary if possible
- List test cases combination of distance and age by Equivalent method
- List test cases combination of distance and age by Boundary method
- Create decision table base on input distance and age

57

Exercise 5: Register Form

58

- A register form allows to create a username and a password and add the account to the database
- Specification:
 - Username: from 3 to 15 characters, no space and special characters including #,\$,@
 - Password: should be 8 digits, first character is not zero
 - Username + Password valid → “Successful Register”
 - Username invalid → “Invalid Username”
 - Password invalid → “Invalid Password”
 - Username & Password invalid → “Invalid Username”
- Using Equivalent Class and Boundary Value Analysis to design test cases

58

Exercise 6

59

- Class `java.util.Array` có một phương thức `Array.sort(pa, from, to)` sắp xếp tăng dần các phần tử trong mảng pa từ `from` đến `to`. Biết phương thức để lấy về chiều dài của mảng là `pa.length`. Sử dụng EC và BVA để xác định các test case cho phương thức `Array.sort(...)`

59

State Transition Testing Kiểm thử trạng thái

60

60

Basic ideas

61

- Applied when an application gives a different output for the same input, depending on what has happened in the earlier state
- The system can be in a (finite) number of different states and the transitions from one state to another are determined by rules
- Have to determine states, events, actions and transitions that should be tested

61

Basic Terms Thuật ngữ cơ bản

62

- The states that the system may occupy
 - ex: a document is closed or opened
- The transitions from one state to another
- The events that cause a transition
 - ex: the action “closing” a document cause the transition form the state “opened” to “closed”
- The actions that result from a transition
 - ex: an error message, a warning ...

62

Design Test Cases

63

- Create a set of test cases such that **all states** are visited at least once.
May miss important transitions.
- Create a set of test cases such that **all events** are triggered at least once.
May miss both states and transitions
- Create a set of test cases such that **all transitions** are exercised at least once. Subsumes (includes) all-states and all-events
 - Stronger: cover all possible pairs of transitions (1-switch coverage)
 - Even stronger: cover all possible triplets of transitions (2-switch coverage)
- Create a set of test cases such that **all paths** are executed at least once.
Subsumes all others. Can be infeasible - consider loops

63

Example 1: ATM machine

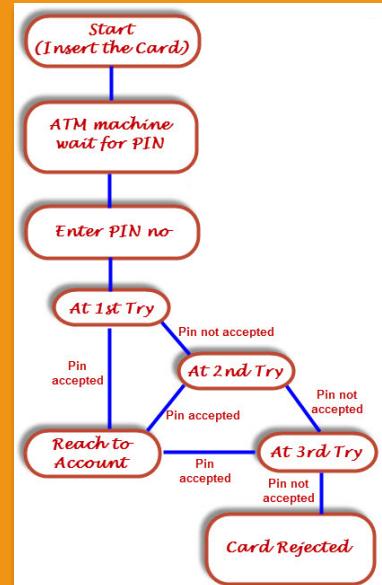
64

- Go to the ATM
- Insert card – Verify PIN
- Withdraw money
 - Successful if sufficient balance
 - Refused if insufficient balance
- User can try to type PIN 3 times – If not valid, card is rejected
- How many states the system has?
- Draw the state-transition diagram

64

State Transition Diagram

- S₁: Start State
- S₂: Wait for PIN
- S₃: 1st try invalid
- S₄: 2nd try invalid
- S₅: 3rd try invalid
- S₆: Card rejected
- S₇: Access account



IT4501 - Software Engineering Department - SoICT/HUST

11/28/22

65

All states coverage

- TC₁: Start State – Valid PIN – Access Account
- TC₂: Start State – 1st try invalid – 2nd try invalid – 3rd try invalid – Card Rejected

	Insert Card	Valid PIN	InValid PIN
Start State	S2	-	-
Wait for PIN	-	S6	S3
1st try invalid	-	S6	S4
2nd try invalid	-	S6	S5
3rd try invalid	-	-	S7
Access Account	-	?	?
Card not excepted	S1 (for new card)	-	-

IT4501 - Software Engineering Department - SoICT/HUST

11/28/22

66

66

All transitions coverage

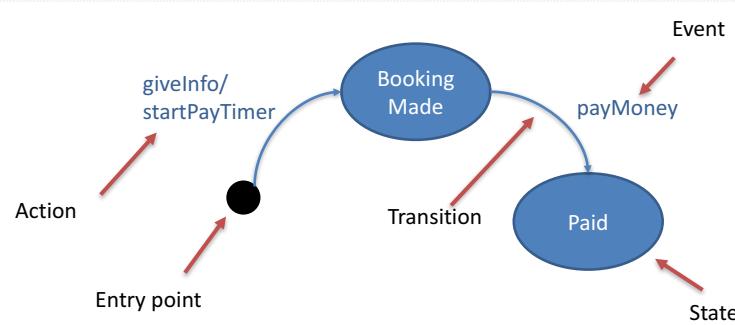
- Which test cases will cover all transitions of the system?

	Insert Card	Valid PIN	InValid PIN
Start State	S2	-	-
Wait for PIN	-	S6	S3
1st try invalid	-	S6	S4
2nd try invalid	-	S6	S5
3rd try invalid	-	-	S7
Access Account	-	?	?
Card not accepted	S1 (for new card)	-	-

Example 2: Flight Booking System

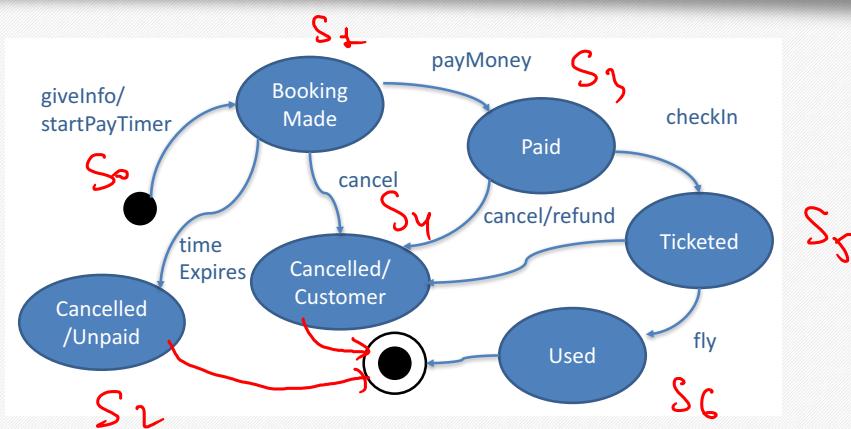
68

- The customer provides some information and makes a booking.
- He then has a certain amount of time to make the reservation.



Flight Booking System Complete

69



69

70

70

Design Test Case for Flight Booking System

71

- Draw a table of states and events
- Identify test cases to satisfy all states coverage
- Identify test cases to satisfy all transitions coverage