SPECIFICATION-BASED TECHNIQUE WORKSHOP Version 1.0

By: Son Pham

Course duration: 3 hours

Last Update: Aug 2015

Global CyberSoft

A World of Difference

www.globalcybersoft.com

Contents

- Overview of Specification-Based techniques
- Exercises for
  - Equivalence Partitioning
  - Boundary Value Analysis
  - Decision Table
  - State Transition
  - Use Case
  - Pairwise

Overview of Specification-Based Techniques

- Purpose:
  - Deeply understand the basic specification-based techniques via exercises
  - Apply them into real project

- Techniques:
  - Equivalence Partitioning
  - Boundary Value Analysis
  - Decision Table
  - State Transition
  - Use Case
  - Pairwise

Equivalence Partitioning (EP)

EP:

- Effectively test the handling of inputs, outputs, internal values and time-related values.
- EP is used to create equivalence classes which are the sets of values that are processed in the same manner.

Applicability

- Apply at any test level
- Applicable to valid and invalid partitions
- Be strongest when used in combination with BVA
- Common used for smoke testing a new build or new release.

Equivalence Partitioning (EP)

Limitations/Difficulties

- The assumption is incorrect and the values in the partition are not handled in exactly the same way
- Select wrong partitions (e.g Negative, Positive, 0)
- ➔ miss defects

Types of Defects

- functional defects in the handling of various data values.

Equivalence Partitioning – Example 1

One of organization's employment applications based on a person's age as follows :

0–15	Don't hire
16-20	Can hire on a part-time basis only
21–55	Can hire as a full-time employee
56–99	Don't hire

Equivalence Partitioning – Example 1

Parameter	Equivalence Class	Representative value
Age (x)	**Valid EC:**	
	vEC1: 0<=x<=15	10
	vEC2: 16<=x<=20	18
	vEC3: 21<=x<=55	30
	vEC4: 56<=x<=99	60
	Invalid EC:	
	iEC1: x<0	-1
	iEC2: x>99	101

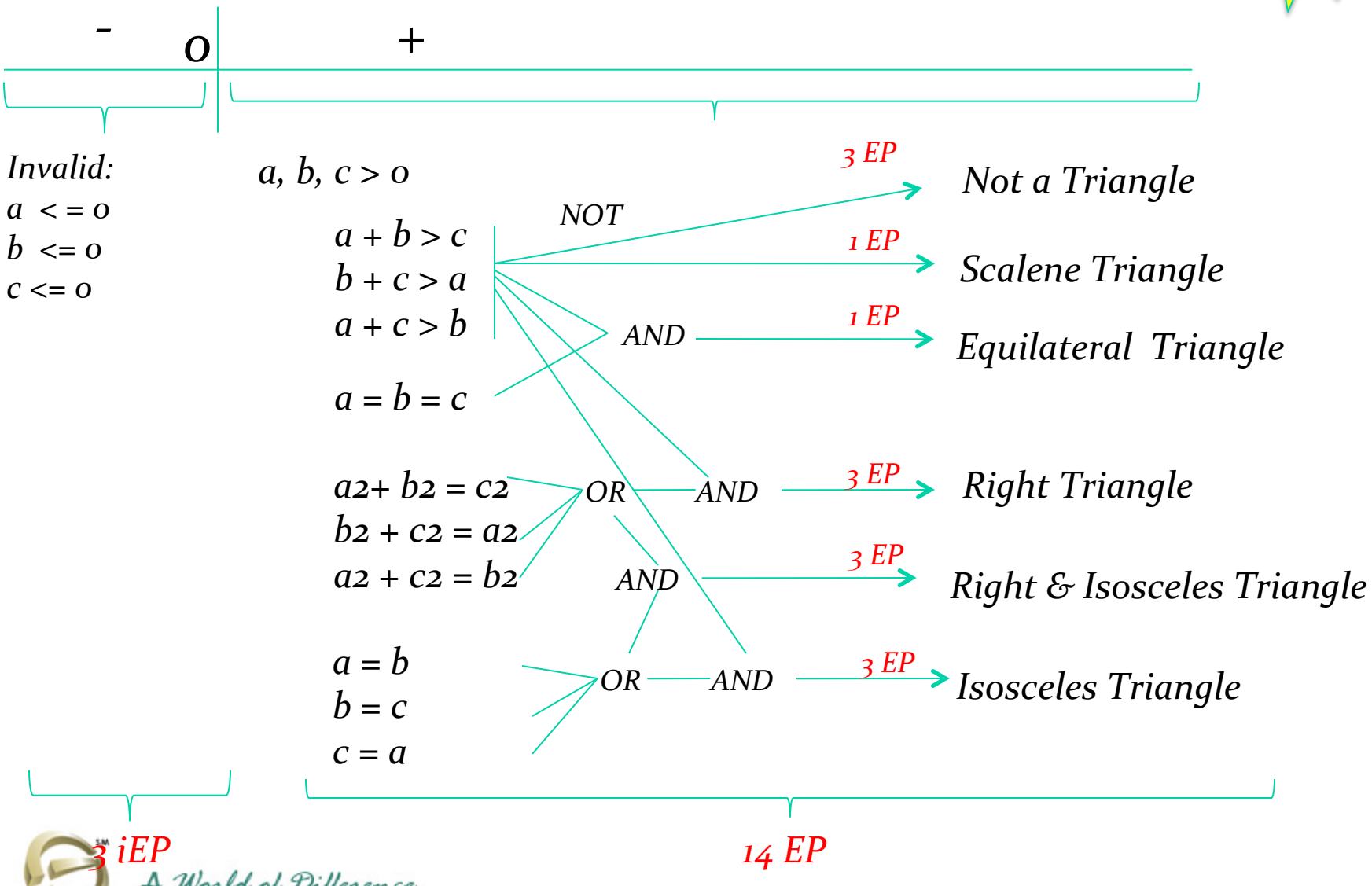
4 valid test cases **2 invalid test cases**

Equivalence Partitioning (EP) – Example 2

Testing a triangle analyzer:
Program specification:
Input: *3 numbers separated by commas or spaces*
Processing:

Determine if three numbers make a valid triangle;
if not, print message NOT A TRIANGLE.
If yes, what is triangle?

Equivalent Partitioning – Example 2



Equivalence Partitioning (EP) – Example 3

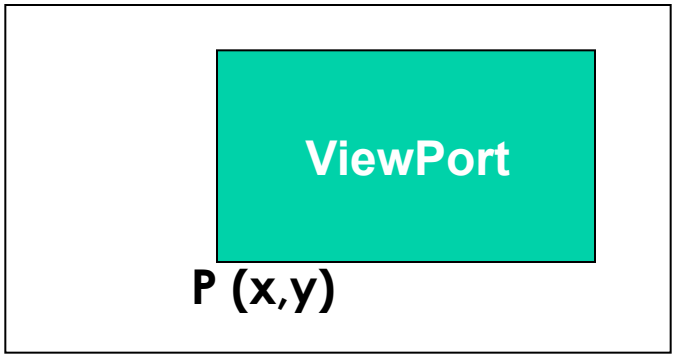
Identify a point (x,y) on ViewPort screen with following information:

x: 230..1000

y: 300..650

→ Identify Test cases

1. Using EP
2. Using BVA



Equivalence Partitioning (EP) – Example 3

Using EC – combine representative values of x and y

Parameter	Equivalence Class	Representative value
x	**Valid EC:**	
	vEC1: 230<=x<=1000	300
	Invalid EC:	
	iEC1: x<230	200
	iEC2: x>1000	1100
y	**Valid EC:**	
	vEC1: 300<=y<=650	400
	Invalid EC:	
	iEC1: y<300	250
	iEC2: y>650	700

1 valid test case **4 invalid test cases.**

Equivalence Partitioning (EP) – Example 4

We are working on a banking project to upgrade an existing automated teller machine system to allow customers to obtain cash advances from supported credit cards. The system should allow cash advances from 20 dollars to 500 dollars, inclusively, for all supported credit cards. The correct list of supported credit cards is American Express, Visa, Japan Credit Bank, Euro Card, and MasterCard.

Create test case by using equivalence partitioning technique.

Boundary Value Analysis (BVA)

BVA:

- To test the values that exist on the boundaries of ordered equivalence partitions
- 2 approaches:
  - 2 values
  - 3 values → being used for the higher risk items

Applicability

- Apply at any test level
- is appropriate when ordered equivalence partitions exist
- Can be applied to the following:
  - Numeric attributes of non-numeric variables (e.g., length) Loops, including those in use cases
  - Stored data structures
  - Physical objects (including memory)
  - Time-determined activities
  - :

Boundary Value Analysis (BVA)

Limitations/Difficulties

- Similar to EP as it depends on the accurate identification of EPs.
- It is not limited to a range of valid inputs.

Types of Defects

- Functional defects
  - displacement or omission of boundaries
  - extra boundaries
- Non-functional defects
  - E.g: tolerance of load limits (e.g., system supports 10,000 concurrent users).

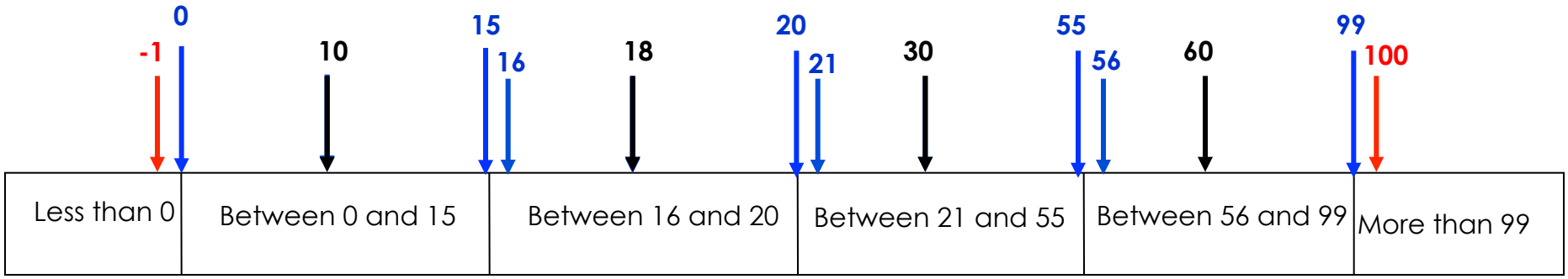
Boundary Value Analysis (BVA) – Example 1

One of organization's employment applications based on a person's age as follows :

0–15	Don't hire
16-20	Can hire on a part-time basis only
21–55	Can hire as a full-time employee
56–99	Don't hire

Boundary Value Analysis – Example 1

- Using EC: 6 EC (4 valid and 2 invalid)
- Using BVA – 2 values: Number of boundary values = 6*2 - 2 = 10



Age – input value

Boundary Value Analysis – Example 1

Age (x)	iEC1	vEC1	vEC2	vEC3	vEC4	iEC2
EC	x<0	0<=x<=15	16<-x<=20	21<=x<=55	56<=x<=99	x>99
EC presentative Data	-1	10	18	30	60	100
BVA	-1	0 , 15	16, 20	21, 55	56, 99	100

➔Using BVA, 10 boundary values: -1, 0, 15, 16, 20, 21, 55, 56, 99, 100

➔Using EC and BVA, we have 14 test cases to cover following values:

-1, 0, 10, 15, 16, 18, 20, 21, 30, 55, 56, 60, 99, 100

Boundary Value Analysis – Example 2

We are testing a financial form. There are two date fields: start date, due date.

- The due date: maximum 90 days, minimum 1 day after the start date.
- Start date, due date must be in 2015

Create test cases using BVA technique.

Boundary Value Analysis – Example 2

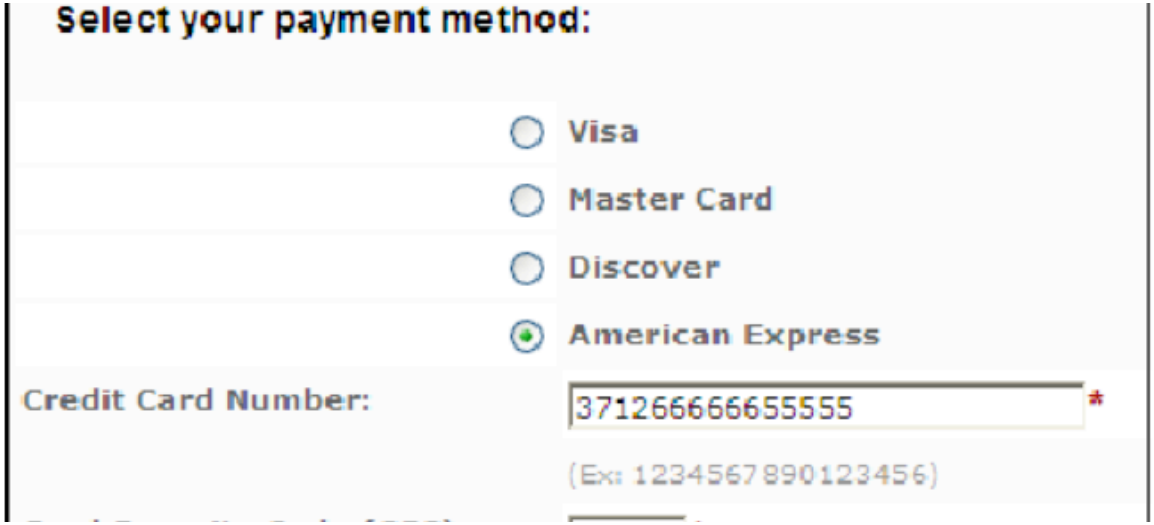
- **Start date:** 31/12/2014, 1/1/2015, 2/1/2015, 30/12/2015, 31/12/2015, 1/1/2016
- **Due date:** 31/12/2014, 1/1/2015, 2/1/2015, 30/12/2015, 31/12/2015, 1/1/2016, due date = start date + 90d, due date = start date + 91d, due date = start date + 89d, due date = start date, due date = start date + 1, due date = start date + 2

	1	2	3	4	5	6	7	8
Start date	**1/1/2015**	**2/1/2015**	**31/12/2015**	**30/12/2015**	1/1/2015	1/10/2015	1/10/2015	**31/12/2014**
Due date	**2/1/2015**	**Start date + 89 d**	31/12/2015	**31/12/2015 (also start date + 1)**	**1/1/2015**	**30/12/2015 (also due date + 90)**	**Start date + 2**	10/1/2015
Accept?	Yes	Yes	No	Yes	No	Yes	Yes	No

	9	10	11	12	13
Start date	**1/1/2016**	10/1/2015	10/12/2015	10/12/2015	10/1/2015
Due date	10/1/2015	**31/12/2014**	**1/1/2016**	**start date**	**Start date + 91**
Accept?	No	No	No	No	No

Boundary Value Analysis – Example 3

We are testing a commercial web application. The payment page is as below:



- Visa, Master, Discovery cards have 16 digits
- American Express cards have 18 digits.
- Credit card field only accepts numbers.
- There will be an error message if credit card number is empty.

- Create test case using equivalence partitioning and BVA techniques.

Boundary Value Analysis – Example 3

Partitions:

Card type: Visa, Master, Discovery, American Express

Credit card number: number, not number

BVA:

Credit card number

Visa, Master, Discovery: 0, 1, 15, 16, 17.

American Express: 0, 1, 17, 18, 19.

	1	2	3	4	5	6	7	8
Card type	**Visa**	**Master**	**Discovery**	**American Express**	Visa	Visa	Master	Discovery
Number	**Number, 16 digits**	Number, 16 digits	Number, 16 digits	**Number, 18 digits**	**Number , 0 digits**	**Number , 1 digit**	**Number, 17 digits**	**Number, 15 digits**
Accept?	Yes	Yes	Yes	Yes	No	No	No	No

	9	10	11	12	13
Card type	American Express	American Express	American Express	American Express	Visa
Number	**Number , 0 digits**	**Number , 1 digit**	**Number, 19 digits**	**Number, 17 digits**	**Not, Number**
Accept?	No	No	No	No	No

Decision Table

Decision Table:

- To ensure that every combination of conditions, relationships and constraints is tested
- 2 methods belong to risk-based
  - collapsed decision table
  - collapsed decision table

Applicability:

- Commonly apply at Integration, System Acceptation test; Depend on code (set of decision logic), it can be applicable for unit test.
- Particularly useful when the requirements are presented in the form of flow charts or tables of business rules.
- Decision tables are also a requirements definition technique.
- Important to consider all rules, business rule, defined condition combination and expected outcomes.
- Boundary value analysis and equivalence partitioning are complementary to the decision table technique.

Decision Table

Limitations/Difficulties

- Finding all the interacting conditions can be challenged in case requirement is not clear or not existed.

Types of Defects

- Defect in requirement
- Functional defect:
  - omissions (there is no information regarding what should actually happen in a certain situation) and contradictions.
  - issues with condition combinations that are not handled or are not handled well.

Decision table – Example 1

- Cellphone : Make a decision table

Condition / Check sum	8	4	2	1	1
Valid Battery	N	Y	Y	Y	Y
Valid Signal	-	N	Y	Y	Y
Valid Credit	-		N	Y	Y
Valid phone number	-	-	-	N	Y
Action					
Recharging	Y	N	N	N	N
Get the Signal	N	Y	N	N	N
Buy Credit	N	N	Y	N	N
Get the correct number	N	N	N	Y	Y
Make call successfully	N	N	N	N	Y

*"-" indicate conditions that aren't reached as part of this rule.

Decision table – Example 2

- Program receives 3 parameters: a, b, c then output if it could from a triangle and the type of triangles: Scalene, isosceles, equilateral.

Create test cases using Decision table techniques .

*"-" indicate conditions that aren't reached as part of this rule.

Decision Table – Example 2

	1	2	3	4	5	6	7	8	9	10	11
C1: a < b + c ?	F	T	T	T	T	T	T	T	T	T	T
C2: b < a + c ?	_	F	T	T	T	T	T	T	T	T	T
C3: c < a + b ?	_	_	F	T	T	T	T	T	T	T	T
C4: a = b?	_	_	_	T	T	T	T	F	F	F	F
C5: a = c?	_	_	_	T	T	F	F	T	T	F	F
C6: b = c?	_	_	_	T	F	T	F	T	F	T	F
A1: Not a triangle	X	X	X								
A2: Scalene											X
A3: Isosceles							X		X	X	
A4: Equilateral				X							
A5: Impossible					X	X		X			

➔ 8 Test cases

State Transition Testing

State Transition Testing:

- test the ability of the software to enter into and exit from defined states via valid and invalid transitions

Applicability

- Apply at any level
- applicable for any software that has defined states and has events that will cause the transitions between those states.

State Transition Testing

Limitations/Difficulties

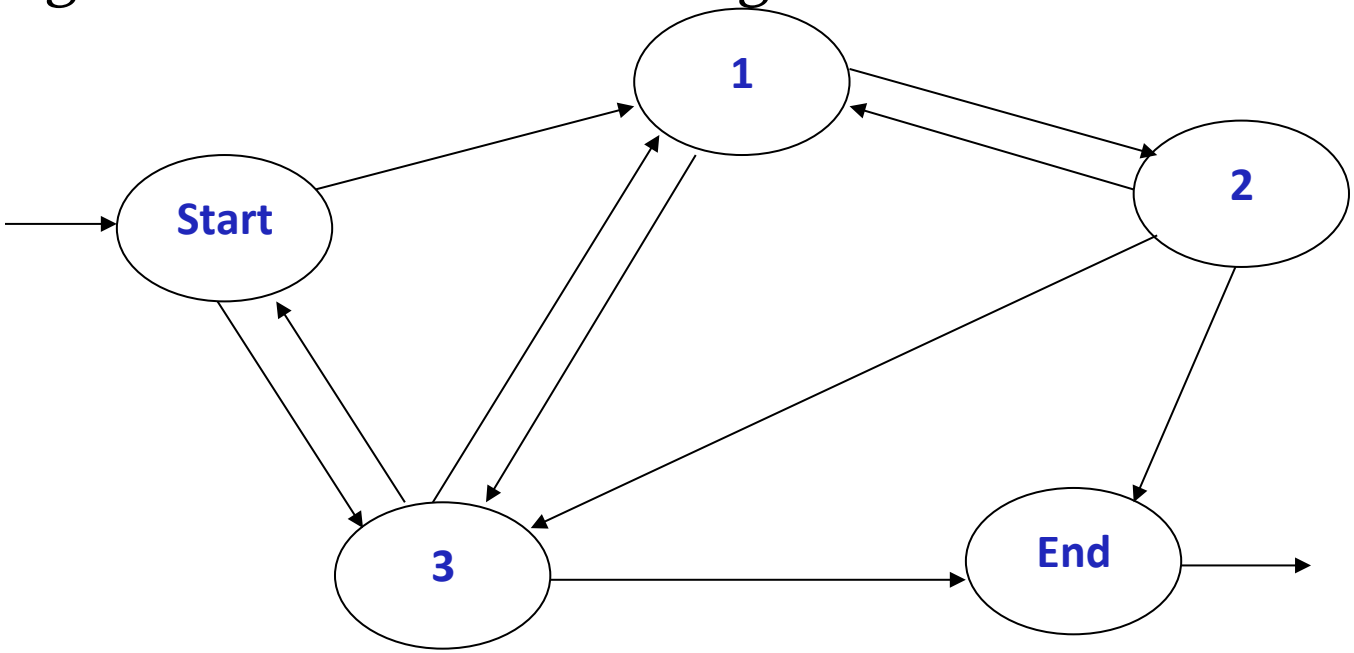
- Difficult to determine the states (E.g: Embedded system)
- Some kinds of state transition defects that found by testing sequences of transactions: 1-Switch, N-Switch

Types of Defects

- Requirement defect
- Functional defect:
  - omissions
  - incorrect processing (incorrect or unsupported transitions, states with no exists)

State Transition Testing – Example 1

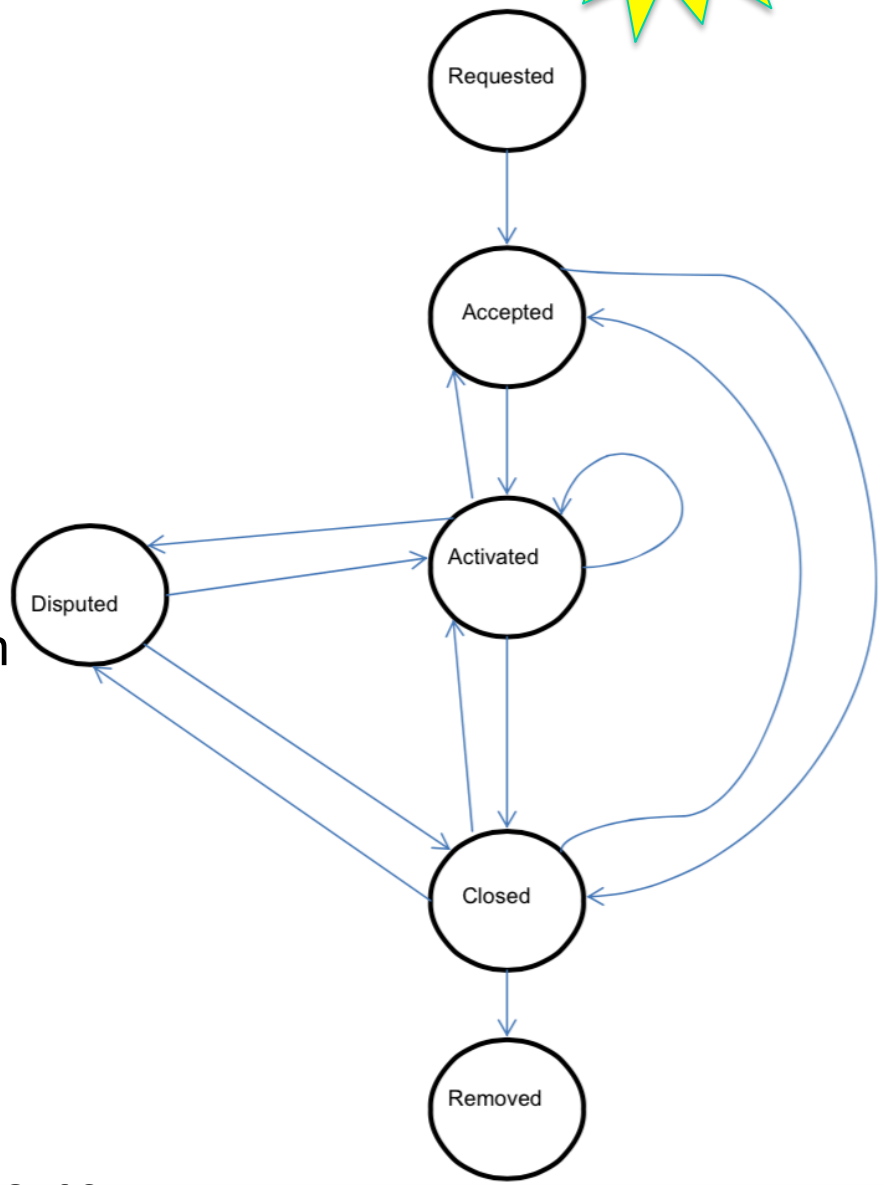
Giving the state transition diagram as below:



1. Identify the number of test cases to cover all states
2. Identify the number of test cases to cover all transitions (0-switch)
3. How many test cases to check the state transition through 1 immediate transition? (Chow–1/1–switch)

State Transition Testing – Example 2

- Employee requests payment of a claim from his/her health insurance.
- If information is missing or changed, the claim may be moved back to an earlier state to prevent payment.
- There are the following restrictions: If a claim in state Accepted has been Closed it can only be restored to the same state Accepted. If a claim in state Activated has been Closed it can only be restored to state Activated.



Starting from Activated: what is the number of 0-switch transitions and what is the number of allowed 1-switch?

Use Case Testing

Use case:

- Use cases are defined in terms of interactions between the actors and the system that accomplish some goal. Actors can be users or external systems.
- Use case testing provides transactional, scenario-based tests that should emulate usage of the system.

Applicability

- Usually applied at the system and acceptance testing levels; can be applied at lower levels depending on behavior of component.
- UC is the basis for performance testing.

Use Case Testing

Limitations/Difficulties

- The value of a use case is reduced if the use case does not accurately reflect activities of the real user.

Types of Defects

- mishandling of defined scenarios,
- missed alternate path handling
- incorrect processing of the conditions presented and awkward or incorrect error reporting.

Use Case Testing – Example 1

USE CASE: ADD TO EASYTRAVEL BALANCE FROM CREDIT CARD

Use case ID: UC-201201

Purpose: User is increasing the balance on their Easytravel card. Actors: user, system

Pre-conditions: User has a valid Easytravel card and a credit card.

Use Case Testing - Example

Basic Flow:

User	System
1. User sets the Easytravel card on the reading plate of the Easytravel Loading Machine.	2. The system asks what the user wishes to do: (E1) a) query card balance (separate use case) b) add to balance of the card c) check latest card transactions (separate use case)
3. User chooses “Add balance”	4. System asks the amount. (E1)
5. User selects the amount.	6. System asks for the payment method: (E1) a) cash (separate use case) b) credit card
7. User selects credit card.	8. System asks the user to insert credit card into the credit card reader. (E1)
9. User inserts the credit card.	10. System shows the amount to be charged from the credit card and asks for confirmation. (E2)
11. User confirms the amount.	12. System makes the credit card transaction and adds the amount to the Easytravel card balance.
13. User removes the credit card and the Easytravel card.	14. System prints out a receipt of the transaction.
	15. System returns to the main screen.

Use Case Testing - Example

Exception	Action
E1	User can stop the process by removing the Easytravel card from the reading plate.
E2	If the user does not accept the amount to be charged, they can cancel the operation by pressing the Cancel-button on the credit card reader.

- End result: User's Easytravel card balance has been increased with the selected amount and the equal amount has been charged to the credit card.

How many test cases are required to achieve the minimum coverage for this use case?

Pairwise Testing – Example 1

Identify TCs to cover the healthy status

→ Using tools (allpairs or Jenny tools)

Age	Smoking?	Disease History	Health Status
<16	No	3 generations	Strong
16..30	Rarely	Parent	Symptom
31..50	Sometime	Father	Illness
> 50	Often	Mother	Critical Illness
	Very often	No	

Pairwise Testing – Example 2

- A company offering house insurance policies has several policy options. They depend on the following factors:
  - Building type: house, semi-detached, apartment building, cottage
  - Material: wood, concrete, brick, mixed
  - Location: city, suburb, countryside, wilderness

- You are testing the system and using the pairwise technique for creating test cases.

Using the pairwise technique, how many test cases are required to achieve 2-wise coverage?

Appendix: Course detail form

Author	Son Pham	**Duration**	3 hours
Category	Theory and Practice	**Type**	Test Analysis and Design

Examination	
Intended Audience	Any software engineer, project leader, project manager who are working in software related projects.
Pre-requisites	All Test engineers who have background about Basic Specification-Based Techniques
Completion criteria for the course	Attendee must join at least 80% of course length.
Criteria for granting training waivers	

Change Log History

DOCUMENT CHANGE LOG					
Revision	**Description**	**Author**	**Date**	**Approved**	**Date**
1.0	First version.	Son Pham	Dec 2015		
This document has been generated from template TL-QMS-005 Revision 3.4					

Thank you

THANK YOU

Inquires regarding the above may be directed to:
Someone, Title, truonghx@gcs-vn.com