TEST CASE WORKSHOP

Software testing practice

By : Nguyen Thi Thanh Truc

Duration : **1.5 hours**

Revision : **1.1**

Contents

- **Introduction**
- **The Document To Dev Test Case**
- **Test Case Template**
- **Black Box Techniques practice**
  - Equivalence Class Partitioning
  - Boundary Value Analysis
  - **Use-Case Based Testing**
  - Combinational Testing
- **Q&A**
- **Reference**
- **Exercise**

Introduction

This training is intended to provide:

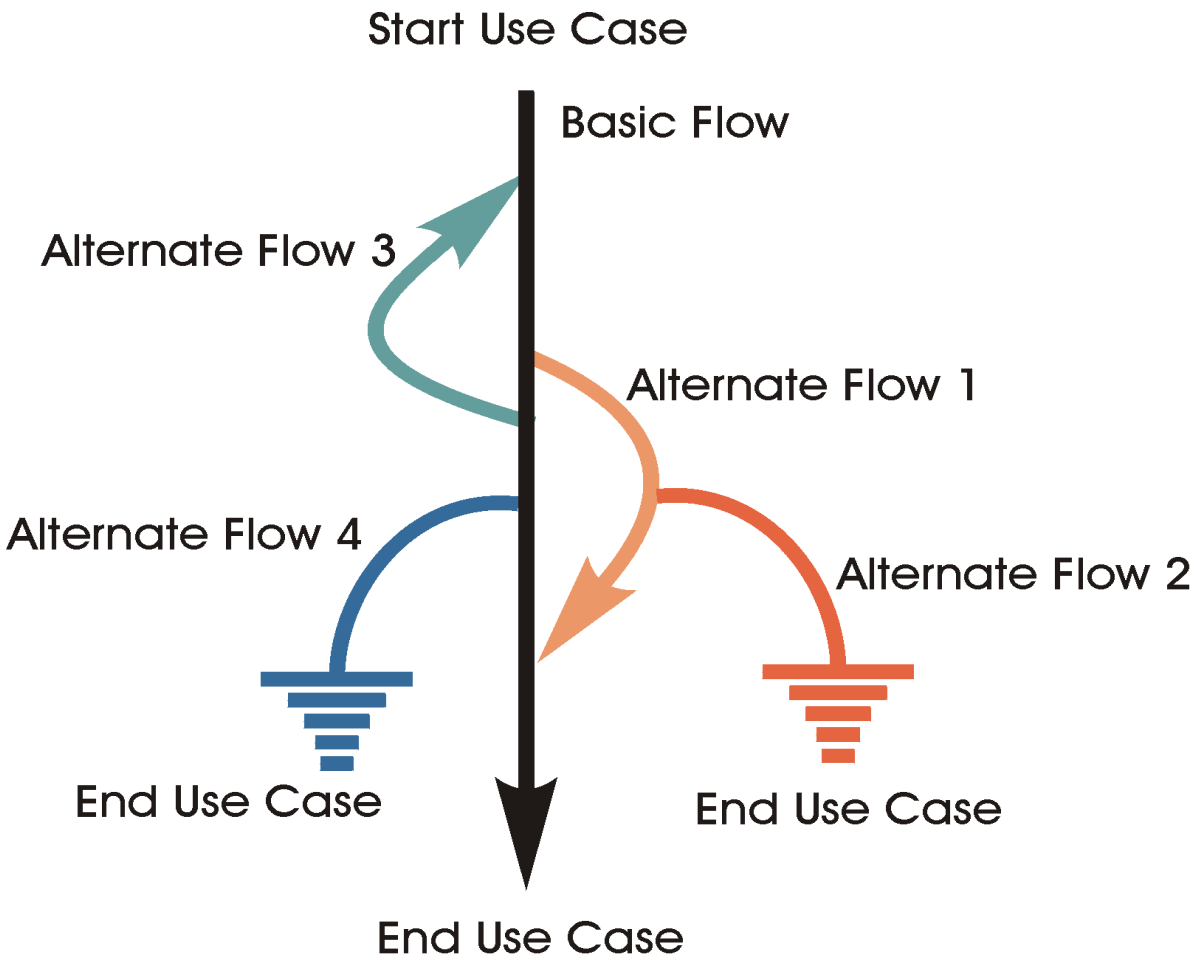
- Documents need to be focused during writing test cases.
- A test case template.
- A practice for some basic techniques:
  - Equivalence Class Partitioning
  - Boundary Value Analysis
  - Use-Case Based Testing
  - Combinational Testing

Use-Case Based Testing

- Test procedure is modeled after the Use Cases which were used to specify and design the system

Example: Base on Flows Of Events – Basic and Alternate Flows:

Use-Case Based Testing (cont)



Use-Case Based Testing (cont)

The different use-case scenarios can be identified by beginning with the basic flow and then combining the basic flow with alternate flows:

Scenario 1	Basic Flow			
Scenario 2	Basic Flow	Alternate Flow 1		
Scenario 3	Basic Flow	Alternate Flow 1	Alternate Flow 2	
Scenario 4	Basic Flow	Alternate Flow 3		
Scenario 5	Basic Flow	Alternate Flow 3	Alternate Flow 1	
Scenario 6	Basic Flow	Alternate Flow 3	Alternate Flow 1	Alternate Flow 2
Scenario 7	Basic Flow	Alternate Flow 4		
Scenario 8	Basic Flow	Alternate Flow 3	Alternate Flow 4	

Use-Case Based Testing - Guideline

- *Deriving the test cases for each scenario is done by identifying the specific condition that will cause that specific use-case scenario to be executed.*

Use-Case Based Testing: Exercises

Basic Flow	This Use Case begins with the ATM in the Ready State. 1. Initiate Withdraw - Customer inserts bank card in the card reader on the ATM machine 2. Verify Bank Card - The ATM reads the account code from the magnetic strip on the bank card and checks if it is an acceptable bank card. 3. Enter PIN - The ATM asks for the customer's PIN code (4 digits) 4. Verify account code and PIN - The account code and PIN are verified to determine if the account is valid and if the PIN entered is the correct PIN for the account. For this flow, the account is a valid account and the PIN is the correct PIN associated with this account. 5. ATM Options - The ATM displays the different alternatives available at this ATM. In this flow, the bank customer always selects "Cash Withdraw." 6. Enter Amount - The ATM the amount to withdraw. For this flow the customer selects a pre-set amount ($10, $20, $50, or $100). 7. Authorization - The ATM initiates the verification process with the Banking System by sending the Card ID, PIN, Amount, and Account information as a transaction. For this flow, the Banking System is online and replies with the authorization to complete the cash withdrawal successfully and updates the account balance accordingly. 8. Dispense - The Money is dispensed. 9. Return Card - The Bank Card is returned. 10. Receipt - The receipt is printed and dispensed. The ATM also updates the internal log accordingly. Use Case ends with the ATM in the Ready State.

Exercises (cont)

Alternate Flow 3 - Insufficient funds in ATM	At Basic Flow Step 6 - Enter Amount, if the ATM contains insufficient funds to dispense the requested amount, an appropriate message will be displayed, and rejoins the basic flow at Step 6 - Enter Amount.

Exercises (cont)

- For instance: Scenario 4 with Alternative Flow 3, the test cases below are developed accordingly:

Test Case ID	Scenario	Condition	Expected Result
TC a	Scenario 4	Step 6 (Basic Flow) - Withdraw Amount > Account Balance	Rejoin basic flow at Step 6
TC b	Scenario 4	Step 6 (Basic Flow) - Withdraw Amount < Account Balance	Does not execute Alternate Flow 3, takes basic flow
TC c	Scenario 4	Step 6 (Basic Flow) - Withdraw Amount = Account Balance	Does not execute Alternate Flow 3, takes basic flow

Write out test cases

- (Self-created)

Combinational Testing

- Many testing situations, there are simply too many test cases to write and execute. So, how do we choose a "good" subset?
- Problem #1:
  - Consider the following situation. Suppose you have a Web site that is hosted on a number of servers and operating systems and viewed on a number of browsers with various plug-ins:
    - Web Browser (Netscape 6.2, IE 6.0, Opera 4.0)
    - Plug-in (None, RealPlayer, MediaPlayer)
    - Application Server (IIS, Apache, Netscape Enterprise)
    - Operating System (Win2000, WinNT, Linux)
  - How many distinct combinations should be tested? The answer is 81 (3x3x3x3 = 81)

Combinational Testing

- Difficulties :
  - What if you don't have enough resources (time, human resources, etc) to do this level of testing?
  - Choosing test cases randomly appears to be not a good solution.
- What needs to be tested ?
  - **At least, all pairs of combinations should be tested :**
  - Each browser is tested with every plug-in, with every server, and with every operating system
  - Each plug-in is tested with every browser, every server, and every operating system.
  - Each server is tested with every browser, every plug-in, and every operating system.
  - Each operating system is tested with every browser, every plug-in, and every server

Combinational Testing - Solution

– A solution :

Test Case	Browser	Plug-In	Server	Operating System
1	Netscape 6.2	None	IIS	Win2000
2	Netscape 6.2	RealPlayer	Apache	WinNT
3	Netscape 6.2	MediaPlayer	Netscape Enterprise	Linux
4	IE 6.0	None	Apache	Linux
5	IE 6.0	RealPlayer	Netscape Enterprise	Win2000
6	IE 6.0	MediaPlayer	IIS	WinNT
7	Opera 4.0	None	Netscape Enterprise	WinNT
8	Opera 4.0	RealPlayer	IIS	Linux
9	Opera 4.0	MediaPlayer	Apache	Win2000

Thank you for your attention!

Q&A