Mahindra Satyam

Day 4

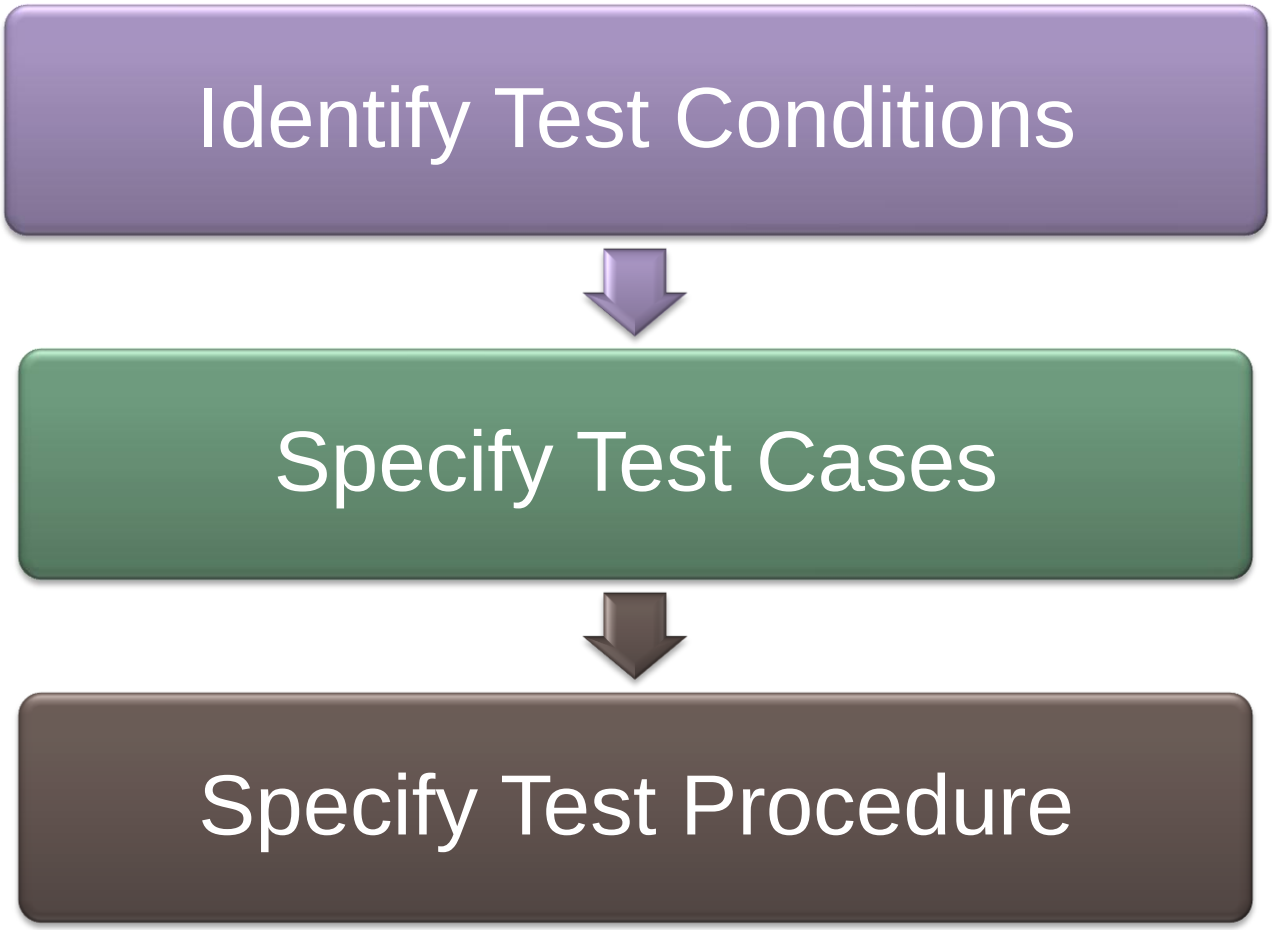
Test Design Techniques

Learning Objectives

- The Test Development Process
- Test Design Techniques
- Specification-based or Black-box test design techniques
  - Test Case Designing
  - Equivalence Partitioning
  - Boundary Value Analysis
  - State Transition Testing

Test Development Process

The design of tests comprises of three main steps:



Contd..

Identify Test Conditions

- An item or event of a component or system that could be verified by one or more test cases.
- e.g. a function, transaction, feature, quality attribute, or structural element.
- Also known as Test Scenarios.

Specify Test Cases

- A set of input values, execution pre-conditions, expected results and execution post-conditions, developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement. [After IEEE 610]

Specify Test Procedure

- A document specifying a sequence of actions for the execution of a test.
- Also known as test script or manual test script.

Test Design Techniques

- Test design technique is a procedure used to derive and/or select test cases.
- Based on deriving test cases the test design techniques are grouped into three categories:
  - ✓ **Specification-based or Black-box techniques**
  - ✓ **Structure-based or White-box techniques**
  - ✓ **Experience-based techniques**

Specification-based or Black-box Techniques

- Procedure to derive and/or select test cases based on an analysis of the specification, either functional or non-functional, of a component or system without reference to its internal structure.
- **Common Characteristics of specification-based test design techniques:**
  - ✓ Models, either formal or informal, are used for the specification of the problem to be solved, the s/w or its component.
  - ✓ Test cases can be derived systematically from these models.
- **There are different types of specification-based techniques, those are:**
  - ✓ **Boundary Value Analysis**
  - ✓ **Equivalence Partitioning**
  - ✓ **State Transition Testing**

Designing Test Cases in General

Test Cases In General

Designing Test Cases

Test Conditions/ Scenarios	Test Cases	Traceability
Test Scenarios Template	Test Case Template	Traceability

Equivalence Partitioning

- **Equivalence Partitioning:**
  - A black box test design technique in which test cases are designed to execute representatives from equivalence partitions. In principle test cases are designed to cover each partition at least once.
- It can be used to achieve input and output coverage goals.
- It can be found for both valid data i.e., values that should be accepted and invalid data, i.e., values that should be rejected.
- Tests can be designed to cover all valid and invalid partitions and it is applicable at all levels of testing.

  - **Example 1: Valid input: Integers in the range 1 to 10.**
    - **# Valid partition:** 1 to 10 inclusive.
    - **# Invalid partitions:** less than 1(blank or zero), more than 10, real (decimal) numbers and non-numeric characters.

Contd..

- **Example 2: Valid input: Names with up to 20 alphabetic characters.**
  - # **Valid partition:** strings of up to 20 alphabetic characters.
  - # **Invalid partitions:** strings of more than 20 alphabetic characters, strings containing non-alphabetic characters.

❖ **Exercise 1:**

➢ A thermometer measures temperature in whole degrees only. If the temperature falls below 18 degrees, the heating is switched off. It is switched on again when the temperature reaches 21 degrees. What are the best values in degrees to cover all equivalence partitions?

❖ **Exercise 2:**

➢ An employee's bonus is to be calculated. It cannot become negative, but it can be calculated to zero. The bonus is based on the duration of the employment. An employee can be employed for less than or equal to 2 years, more than 2 years but less than 5 years, 5 to 10 years, or longer than 10 years. Depending on this period of employment, an employee will get either no bonus or a bonus of 10%, 25% or 35%. How many equivalence partitions are needed to test the calculation of the bonus?

Boundary Value Analysis

Boundary Value Analysis:

- A black box test design technique in which test cases are designed based on boundary values.

- This technique is used to test the behavior at the edge of each equivalence partition because boundaries are an area where testing is likely to yield defects.
- The maximum and minimum values of a partition are its boundary values.
- A boundary value for a valid partition is a valid boundary value; the boundary of an invalid partition is an invalid boundary value.
- Test can be designed to cover both valid and invalid boundary values.
- Boundary Value Analysis can be applied at all test levels.
- It is easy to apply and its defect finding capacity is high.

Contd..

Example 1:

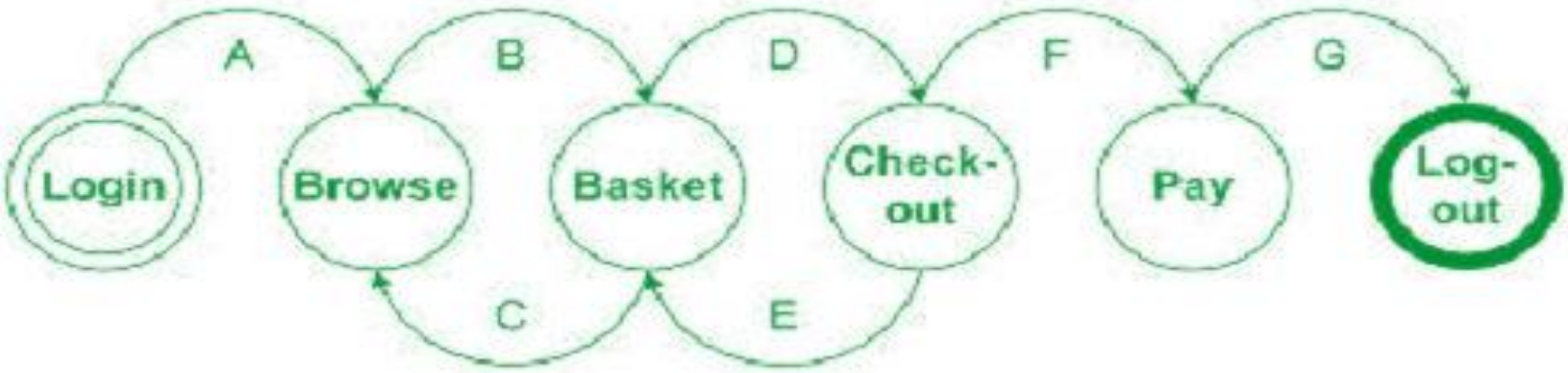
- If an exam has a pass boundary at 40 percent, merit at 60 percent and distinction at 80 percent.
  - The 3 value boundaries would be 39, 40, 41 for pass, 59, 60, 61 for merit, 79, 80, 81 for distinction.
  - The 2 value equivalents would be 39 and 40, 59 and 60, 79 and 80 respectively.

Example 2:

- A wholesaler sells printer cartridges. The minimum order quantity is 5. There is a 20% discount for orders of 100 or more printer cartridges. You have been asked to prepare test cases using various values for the number of printer cartridges ordered. Which of the following groups contain three test inputs that would be generated using Boundary Value Analysis?

State Transition Testing

- **State Transition Testing:**
  - A black box test design technique in which test cases are designed to execute valid and invalid state transitions.
- A state table shows the relationship between the states and inputs, and can highlight possible transactions that are invalid.
- Tests can be designed to cover a typical sequence of states, to cover every state, to exercise every transition, to exercise specific sequences of transitions or to test invalid transitions.
- **Example**

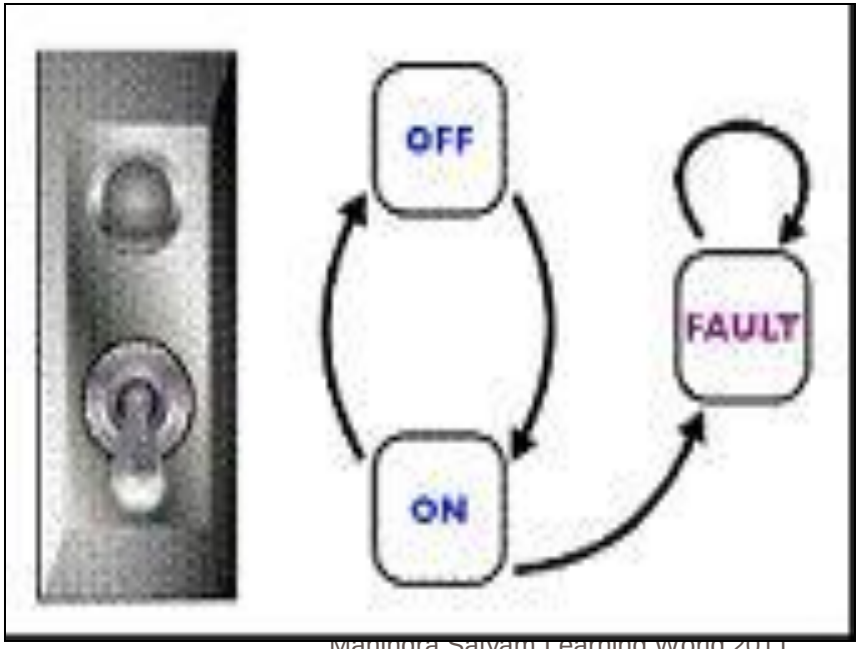


Contd..

Exercise 1:

Consider the following state transition diagram of a switch. Which of the following represents an invalid state transition?

A). OFF to ON
B). ON to OFF
C). FAULT to ON
D). ON to FAULT
E). FAULT to FAULT



White-box/Structure-based techniques

- White-box test design techniques are based on an analysis of the structure of the component or system.
- Component Level: The structure of a s/w component, i.e., statements, decisions, branches or even distinct paths.
- Integration Level: The structure may be a call tree. (a modules calls other modules)
- System Level: The structure may be a menu structure, business process or web page structure.

- **Common Characteristics of structure-based test design techniques:**
  - Information about how the s/w is constructed is used to derive the test cases (e.g., Code and detail design information)
  - The extent of coverage of the s/w can be measured for existing test cases, and further test cases can be derived systematically to increase coverage.

Statement Testing and Coverage

- Code can be of two types, executable and non-executable.
- Executable code instructs the computer to take some action; non-executable code is used to prepare the computer to do its calculations but it does not involve any actions.
  - Ex: Reserving space to store a calculation involves no actions.
- In pseudo code non-executable statements will be at the beginning of the program; the start of the executable part is usually identified by BEGIN, and the end of the program by END.

  So we get the following structure:

  1 Non-executable statements
  2 BEGIN
  3
  4 Executable statements
  5
  6 END

- If we were counting executable statements we would count lines 2, 4 and 6.
- Line 1 is not counted because it is non-executable. Lines 3 and 5 are ignored because they are blank.

Contd..

- The statement testing technique derives test cases to execute specific statements, normally to increase statement coverage.
- In component testing, statement coverage is the assessment of the percentage of executable statements that have been exercised by a test case suite.
- **Statement Coverage is determined by:** No. of executable statements covered by test cases divided by No. of all executable statements in the Code.
- If we aim to test every executable statement we call this full or 100 per cent statement coverage.
- If we exercise half the executable statements this is 50 per cent statement coverage, and so on.
- **Remember:** we are only interested in executable statements, so we do not count non-executable statements at all when we are measuring statement coverage.

Decision Testing and Coverage

- Decision testing aims to ensure that the decisions in a program are adequately exercised.
- Decision testing technique derives test cases to execute specific decision outcomes.
- **Decision coverage is determined by:** the number of all decision outcomes covered by test cases divided by the number of all possible decision outcomes in the code under test.
- Decision coverage is stronger than statement coverage. 100% Decision coverage guarantees 100% statement coverage, but not vice versa.
- **Let us consider the example:**
  - A test case that sets the variable Count to 5 and then inputs the values 1, 5, −2, −3, 6 will exercise all the decisions fully and provide us with 100 per cent decision coverage.

```
1 Program Check
2
3 Count, Sum, Index: Integer
4
5 Begin
6
7 Index = 0
8 Sum = 0
9 Read (Count)
10 Read (New)
11
12 While Index <= Count
13 Do
14 If New < 0 Then
15 Sum = Sum + 1
16 Endif
17 Index = Index + 1
18 Read (New)
19 End Do
20
21 Print ("There were", Sum, "negative numbers in the input stream")
22
23 End
```

Experience-based test design techniques

- Experience-based testing is where tests are derived from the tester's skill and intuition and their experience with similar applications and technologies.
- It is an approach that is most useful where there are few or inadequate specifications and severe time pressure.
- Black-box and White-box testing may also be combined with experience-based techniques to leverage the experience of developers, testers, and users to determine what should be tested.
- **Common Characteristics of experience-based test design techniques:**
  - The knowledge and experience of people are used to derive the test cases.
  - The knowledge of testers, developers, users and other stakeholders about the s/w, its usage and its environment is one source of information.
  - Knowledge about likely defects and their distribution is another source of information

Contd..

Different types of experience-based testing techniques:

- **Error Guessing:** A structured approach to the error guessing technique is to enumerate a list of possible defects and to design tests that attack these defects. This systematic approach is called fault attack.
- These defect and failure lists can be built based on experience, available defect and failure data, and from common knowledge about why s/w fails.
- **Exploratory Testing:** It is a technique that combines the experience of testers with a structured approach to testing where specifications are either missing or inadequate and where there is severe time pressure.
- It exploits concurrent test design, test execution, test logging and learning within time-boxes and is structured around a test charter containing test objectives.
- In this way exploratory testing maximizes the amount of testing that can be achieved within a limited time frame, using test objectives to maintain focus on the most important areas.

Choosing Test Techniques

The choice of which test techniques to use depends on:

- ✓ Type of system
- ✓ Regulatory standards
- ✓ Customer or contractual requirements
- ✓ Level of risk
- ✓ Type of risk
- ✓ Test objectives
- ✓ Documentation available
- ✓ Knowledge of the testers
- ✓ Time and budget
- ✓ Development life cycle
- ✓ Use case models
- ✓ Previous experience with types of defects found

Thank you

mahindrasatyam.net

Safe Harbor

This document contains forward-looking statements within the meaning of section 27A of Securities Act of 1933, as amended, and section 21E of the Securities Exchange Act of 1934, as amended. The forward-looking statements contained herein are subject to certain risks and uncertainties that could cause actual results to differ materially from those reflected in the forward-looking statements. Satyam undertakes no duty to update any forward-looking statements. For a discussion of the risks associated with our business, please see the discussions under the heading “Risk Factors” in our report on Form 6-K concerning the quarter ended September 30, 2008, furnished to the Securities and Exchange Commission on 07 November, 2008, and the other reports filed with the Securities and Exchange Commission from time to time. These filings are available at http://www.sec.gov