SNJB's KBJ College of Engineering

Chandwad–423101 (Nashik)

Department of Information Technology

Subject : Software Testing & Quality Assurance (STQA) of BE 2015 Pattern

UNIT – II
TESTING TECHNIQUES AND LEVELS OF TESTING

Using White Box Approach to Test Design

- Tester’s goal is to determine if all logical and data elements in software unit are functioning properly
- This is white/glass box, approach to test case design
- Knowledge needed for white box test design often becomes available to tester in later phases of SDLC
- Specifically during design phase of development
- White box-based test design is most useful when testing small components

Code Functional Testing

- Functional testing is quality assurance (QA) process
- Type of black-box testing that bases its test cases on specifications of software component under test
- Functions are tested by feeding them input and examining the output
- Internal program structure is rarely considered unlike white-box testing
- Usually describes *what* the system does
- Tests a slice of functionality of the whole system

Code Functional Testing

- Functional testing typically involves six steps
  - Identification of functions that the software is expected to perform
  - Creation of input data based on function's specifications
  - Determination of output based on function's specifications
  - Execution of the test case
  - Comparison of actual and expected outputs
  - To check whether the application works as per the customer need

Using Black Box Approaches to Test Case Design

- Considering only inputs and outputs as a basis for designing test cases
- How do we choose a suitable set of inputs from the set of all possible valid and invalid inputs?
- Infinite time and resources are not available to exhaustively test all possible inputs
- This is prohibitively expensive even if the target software is a simple software unit
- Number of test cases would rise rapidly to point of infeasibility

Using Black Box Approaches to Test Case Design

- Goal for smart tester is to effectively use resources available by developing a set of test cases that gives maximum yield of defects for time and effort spent
- To help achieve this goal using the black box approach we can select from several methods
- Very often combinations of the methods are used to detect different types of defects
- Some methods have greater practicality than others

Random Testing

- Each software system has an input domain from which test input data is selected
- If tester randomly selects inputs from the domain, this is called random testing
- Use of random test inputs may save some of the time and effort
- Selecting test inputs randomly has very little chance of producing an effective set of test data
- There are some tools that generate random test data for stress tests

Requirements Based Testing

- Testing approach in which test cases, conditions and data are derived from requirements
- Include functional tests & non-functional attributes such as performance, reliability or usability
- Stages in Requirements Based Testing:
  - Defining Test Completion Criteria
  - Design Test Cases
  - Execute Tests
  - Verify Test Results & Test Coverage
  - Track and Manage Defects

Decision Tables

- Deal with combinations of things (e.g. inputs)
- Also referred to as a cause-effect table
- Provide systematic way of stating complex business rules, which is useful for developers and testers
- Can be used in test design whether or not they are used in specifications
- Testing combinations can be a challenge, as the number of combinations can often be huge
- Testing all combinations may be impractical

How to Use decision tables for test designing?

- First task is to identify suitable function which reacts according to combination of inputs
- System should not contain too many inputs otherwise number of combinations will become unmanageable
- It’s better to deal with large numbers of conditions by dividing them into subsets and dealing with the subsets one at a time
- Once you have identified aspects that need to be combined, then you put them into a table listing all combinations of *True & False* for each of aspects

Decision Table for Credit Card Example

Conditions	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
New customer (15%)	T	T	T	T	F	F	F	F
Loyalty card (10%)	T	T	F	F	T	T	F	F
Coupon (20%)	T	F	T	F	T	F	T	F
Actions								
Discount (%)	X	X	20	15	30	10	20	0

State-based Testing

- New method for testing object-oriented programs
- Information stored in state of object is of two kinds:
  - Control-information
  - Data-storage
- Validates expected transformations that can occur within a class
- Based on modelling of class as finite state automata
- Automata are used to generate the test cases

State-based Testing

- Each test cases involves
  - Creation of specific state for object
  - Invocation of an operation
  - Concluded by validation of final state achieved by object

Cause-effect Graphing

- Directed graph that maps a set of causes to a set of effects
- Causes may be the input to the program, and the effects may be the output
- Usually graph shows nodes representing causes on left side & nodes representing effects on right side
- There may be intermediate nodes in between that combine inputs using logical operators such as AND and OR

Cause-effect Graphing

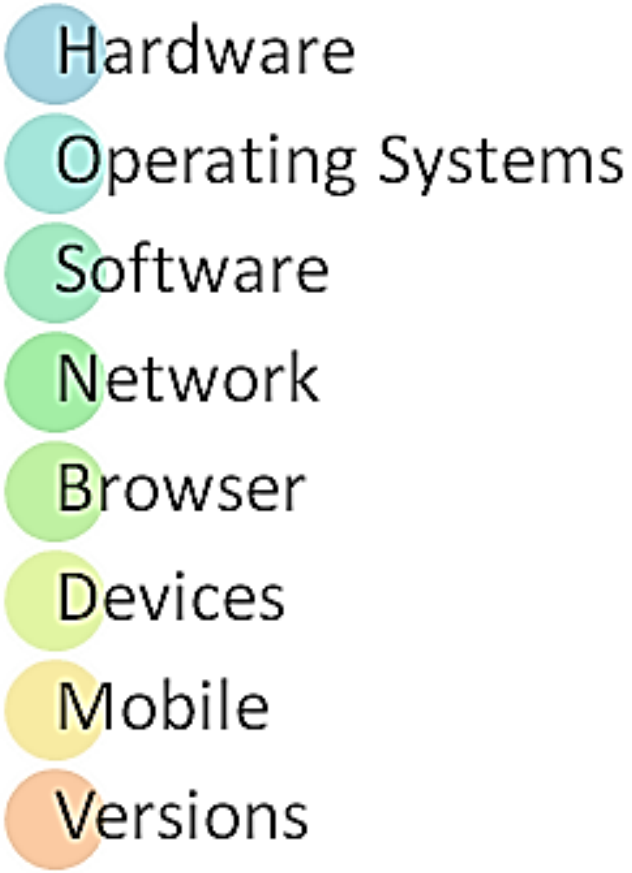
- Directed graph that maps a set of causes to a set of effects
- Causes may be the input to the program, and the effects may be the output
- Usually graph shows nodes representing causes on left side & nodes representing effects on right side
- There may be intermediate nodes in between that combine inputs using logical operators such as AND and OR

Error Guessing

- Based on the tester's/developer's past experience with code similar to the code-under-test And their perception as to where defects may lurk in code
- Tester sometimes able to make an educated guess as to which types of defects may be present and design test cases to reveal them
- Some examples are division by zero, conditions around array boundaries, etc.
- Error guessing is an ad hoc approach to test design in most cases

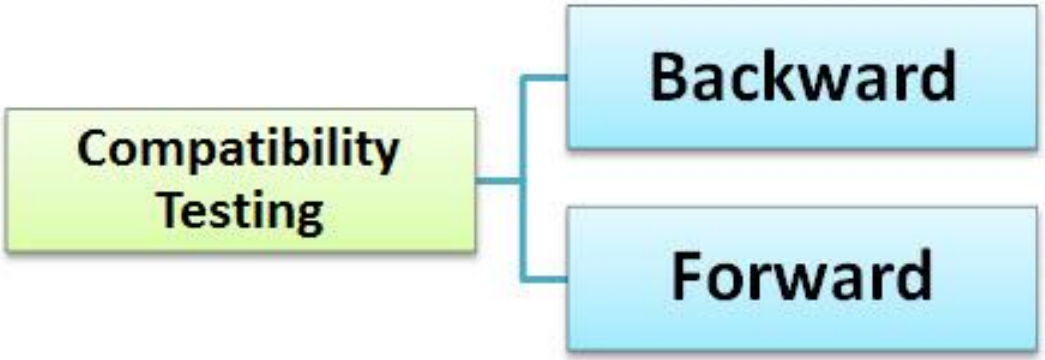
Compatibility Testing

- Type of Software testing to check whether your software is capable of running on →
- Type of the Non-functional testing



Compatibility Testing

- There are two types of version checking



- **Backward** verifies behavior of developed hardware/software with **older versions** of hardware/software
- **Forward** – new version

Compatibility Testing

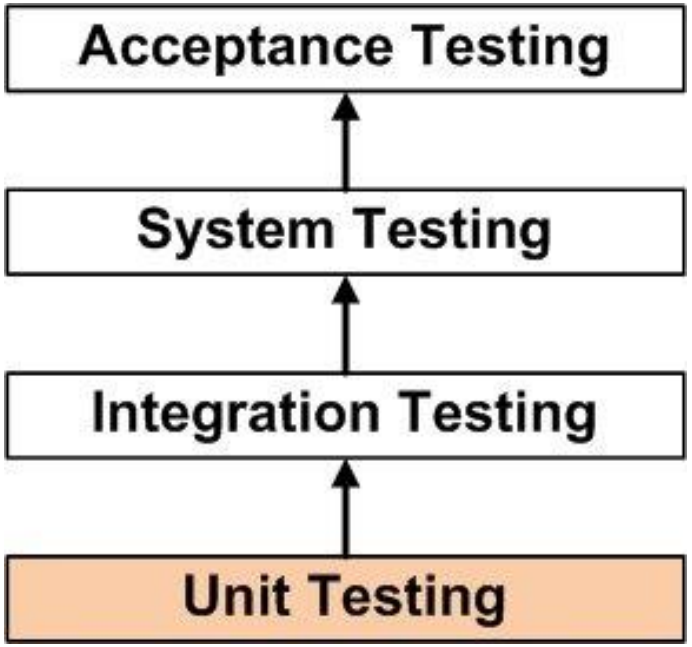
- Tools for Compatibility Testing
  - BrowserStack - Browser Compatibility Testing
    - This tool helps us to check your application in different browsers
  - Secure Platform - Hardware Compatibility tool
    - This tool includes necessary drivers for a specific hardware platform and it provides information on a tool to check for CD burning process with CD burning tools
  - Virtual Desktops - Operating System Compatibility
    - This is used to run the applications in multiple operating systems as virtual machines. Number of systems can be connected and compare the results

Levels of Testing - Unit Testing

- Level of software testing where individual units/ components of a software are tested
- Purpose is to validate that each unit of the software performs as designed
- A unit is the smallest testable part of any software
- It usually has one or a few inputs and usually a single output
- In procedural programming, a unit may be an individual program, function, procedure, etc

Levels of Testing - Unit Testing

- In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class



Levels of Testing - Unit Testing

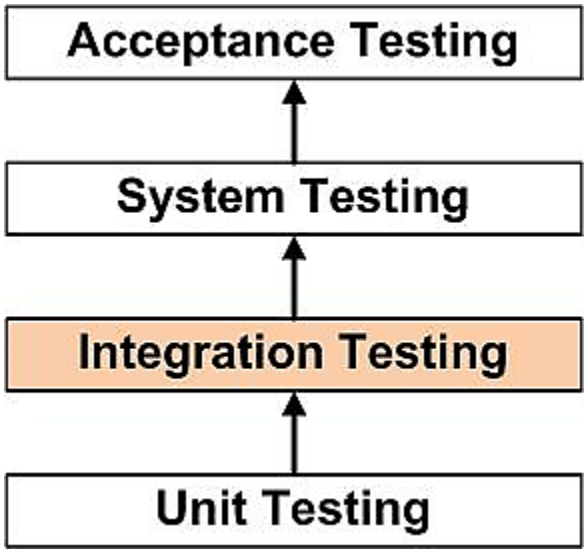
- First level of software testing and is performed prior to Integration Testing
- Performed by software developers themselves
- In rare cases, it may also be performed by independent software testers

Integration Testing

- Level of software testing where individual units are combined and tested as a group
- Purpose of this level of testing is to expose faults in the interaction between integrated units
- Second level of testing performed after Unit Testing and before System Testing
- Developers themselves or independent testers perform this testing

Integration Testing

- Works best as an iterative process in procedural oriented system
- One unit at a time is integrated into a set of previously integrated modules



Defect Bash Elimination

- Defect bash is testing event where group of people tries to find as many defects as possible
- Testing by all the participants during defect bash is not based on written test cases
- What is to be tested is left to an individual's decision and creativity
- They can also try some operations which are beyond the product specifications
- Widely applied in software companies

Defect Bash Elimination

- Brings together plenty of good practices
  - Enabling people “Cross boundaries and test beyond assigned areas”
  - Bringing different people performing different roles together in the organization for testing
  - Letting everyone in the organization use the product before delivery
  - Bringing fresh pairs of eyes to uncover new defects
  - Enabling people to say “system works” as well as enabling them to “break the system”

Configuration Testing

- Allows testers to evaluate system performance and availability when hardware exchanges and changes in configurations occur
- Typical software systems interact with hardware devices such as disc drives, tape drives and printers
- Many software system also interact with many CPU
- In many cases, users require that devices be interchangeable, removable, or reconfigurable
- Some software will have set of commands or menus that allows user to make these configuration changes

Configuration Testing

- According to Beizer, configuration testing has the following objectives:
  - Show that all the configuration changing commands and menus work properly
  - Show that all interchangeable devices are really interchangeable, and that they each enter the proper states for specified conditions
  - Show that the systems‘ performance level is maintained when devices are interchanged, or when they fail

Usability Testing

- Technique used in user-centered interaction design to evaluate product by testing it on users
- Irreplaceable usability practice, since it gives direct input on how real users use the system
- Focuses on measuring a human-made product's capacity to meet its intended purpose
- Examples are food, consumer products, web sites or web applications, computer interfaces, etc.

Accessibility Testing

- Subset of Usability Testing
- Performed to ensure that application being tested is usable by people with disabilities like hearing, color blindness, old age, etc.
- Accessibility aim to cater people of different ability:
  - Visual Impairments
  - Physical Impairment
  - Hearing Impairment
  - Cognitive Impairment
  - Learning Impairment

Accessibility Testing

- A good web application should cater to all sets of people and NOT just limited to disabled people
  - Users with poor communications infrastructure
  - Older people and new users, who are often computer illiterate
  - Users using old system (NOT capable of running the latest software)
  - Users, who are using NON-Standard Equipment
  - Users, who are having restricted access

Accessibility Testing

- **Speech Recognition Software -** Convert spoken word to text , which serves as input to computer
- **Screen reader software** - Used to read out the text that is displayed on the screen
- **Screen Magnification Software**- Used to enlarge the monitor and make reading easy for vision-impaired users
- **Special keyboard** made for the users for easy typing who have motor control difficulties

Static Testing Vs. Structural Testing

- Static Testing
  - Technique by which we can check the defects in software without actually executing it
  - Its counter-part is Dynamic Testing which checks an application when code is run
  - Done to avoid errors at early stage of development
  - Helps find errors that may not be found by Dynamic Testing
  - Two main types of static testing
    - Manual Examinations
    - Automated Analysis using Tools

Static Testing Vs. Structural Testing

- Structural Testing
  - Also known as glass box testing or white box testing
  - Tests are derived from the knowledge of the software's structure or internal implementation
  - Other names of structural testing includes clear box testing, open box testing, logic driven testing or path driven testing