UNIT II

Fundamentals of Software Testing

Contents

Review of Fundamentals of Software Testing

- **What is software Testing?**
- ➢“Execution of work product with intent to find defects”
  - ➢It is the process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements
- **Process**
  - Sequence of steps performed for a given purpose
- **Software Process**
  - A set of activities, methods, practices and transformations that people use to develop and maintain software and associated products

Testing Objectives

- **Primary:**
  - ➢ Execute a program with a intent of finding errors to
    - Determine whether system meets specification
    - Determine whether system meets user’s needs
- **Secondary:**
  - Continuously improve the testing process

Testing Principles

- Define expected output or result for each test case executed, to understand if expected or actual output matches or not.
- Developers must not test their programs. Blindfolds cannot be removed in self testing
- Inspect the result of each test completely and carefully
- Include the test cases for invalid and unexpected conditions which are feasible during production
- Test the program to see if it does what it is not supposed to do as well as what is supposed to do.
- Reusability of test case is important for regression
- Does not plan test assuming that no errors will be found
- The probability of locating more errors in any one module is directly proportional to the number of errors already found in the module

Evolution of Software Testing

- **Debugging Oriented Testing**
  - Testing considered as part of software development
  - Not documented
- **Demonstration Oriented Testing**
  - An introduction of software testers independent of development activity
  - Main aim to show software really works
- **Destruction Oriented Testing**
  - Software does not fail at some abnormal instances
- **Evaluation Oriented Testing**
  - Product as well as process of software development is evaluated
- **Prevention Oriented Testing**
  - Testing considered as a prevention activity

Why Testing?

- Provide confidence in the system
- Identify areas of weakness
- Establish the degree of quality
- Establish the extent that the requirements have been met, i.e. what the users asked for is what they got not what someone else though they wanted
- To provide an understanding of the overall system
- To prove it is both usable and operable
- To provide sufficient information to allow an objective decision on applicability to deploy

Stakeholders View of Testing

- **Manager’s View of Software Testing**
  - Product must exactly meet customer expectations
  - Product must be safe & reliable during use
  - Processes used for development and testing must capable of finding defects
- **Tester’s View of Software Testing**
  - Discover defects in the product and process related to development
  - Discover every conceivable fault or weakness in work product
- **Customer’s View of Software Testing**
  - Must be able to find all possible defects in software, along with related documentation
  - Must ensure any legal /regulatory requirements are compiled

Approaches of Testing

- **Big Bang Approach**
  - ➢ Testing software system after development work is completed
  - ➢ This is also termed system testing or final testing
  - ➢ Testing done before release of the product
  - ➢ Main thrust on black box testing
  - ➢ Testing done at end of development cycle may show the defect at any phase of development
  - ➢ Characterized by huge rework, retesting, scrap, sorting of software component and programs after the product built.
  - ➢ Regression testing reveals many issues in product

Approaches of Testing

- **TQM Approach**
  - Process definition for testing
  - Processes are optimized and capable
  - No defect when it is deliver to customer
  - **Stage 0 –** The cost involved is zero
    - No Validation and verification required
    - Quality is free
  - **Stage 1 –** Defect produced during stage of development
    - small cost of verification and fixing
    - Appraisal cost represented by “10”
  - **Stage 2** – Cost of validation and subsequent defect fixing is much higher than verification. Cost represented by “100”
  - **Stage 3** – Highest cost associated with defect found by customer during acceptance testing. Cost represented by “1000”

Testing During Development Life Cycle

- **Requirement Testing**
  - ➢ It involve mock running of future application using the requirement statements
  - ➢ Evaluates whether all requirements covered in requirement statement or not
  - ➢ Similar to building use cases from requirement statement.
- **The characteristics of requirement verification or review :**
  - ➢ Completeness of requirement statement as per organizations standards
  - ➢ Clarity about what is expected by user at each step of working
  - ➢ Measurability of expected results , possibly in numerals , so that these results can be tested.

Testing During Development Life Cycle

- **Design Testing**
  - It involves testing of high level as well as low level design
  - Talks about reviewing the design by subject matter experts
  - Ensures design meet their exit criteria.
- **The characteristics of requirement verification or review :**
  - Completeness of design in terms of covering all possible outcomes
  - Clarity of flow of data within an application and between different applications
  - Testability of a design which talks about software structure
  - Traceability with requirements

Testing During Development Life Cycle

- **Code Testing**
  - ➢Readable and maintainable in future. (adequate comments)
  - ➢Testable in unit testing
  - ➢Traceable with requirement and design
  - ➢Extra ans missing can be considered as defect
  - ➢Testable in integration and system testing
  - ➢Optimized to ensure better working of product

Testing During Development Life Cycle

- **Test Scenario and test case testing**
  - Test scenario should be clear and complete
  - Represent end-to-end relationship
  - Test scenario should cover all requirements
  - Test scenarios may be prioritized as per requirements
  - Scenarios should be feasible so that they can be constructed during testing
  - Test case should cover all scenarios completely

Requirement Tractability Matrix

- Complete traceability of the software application from requirements through designs and code files up to test scenario, test data, test cases and test results
- Way of doing complete mapping of software
- One can expect Blueprint of an entire application using RTM
-

Advantages of RTM

- Entire SDLC can be tracked completely through RTM
- Any test case failure can be tracked through requirement, design coding etc.
- Any chages in requirements can be affected through entire work product upto test cases and vise versa
- The application become maintainable as one has to complete relationship from requirement till test results

Problems in RTM

- Number of requirements are huge
- Difficult to create RTM Manually
- There may be one to one, one to many and many to many relationship between various elements, maintaining relationship need huge efforts
- Requirement changes frequently, one need to update RTM whenever there is change
- Customer may not find value to it.

Requirement Traceability Matrix

- **Horizontal Traceability**
- An application can be traced from requirement through design and coding till cases and test scenario.
- **Bidirectional Traceability**
- One must go in any direction from any point in traceability
- **Vertical Traceability**
- Traceability may exist in individual column as requirements have inter-dependencies between them
- **Risk Traceability**
- The risks are traced tp requirements are mainly to design which defines control mechanism to reduce probability and impact of risk

Essentials of Software Testing

- Software testing also viewed as an exercise of doing SWOT analysis of software where organization build software on the basis of strength of process for development and testing.

- STRENGTH
  - ➢ Some areas in software are very strong that no defect found during testing in these areas
  - WEAKNESS
    - ➢ The areas in software where requirement compliance is on verge of failure may represent weak areas

Essentials of Software Testing

- OPPORTUNITY
  - ➢ Some areas of software which satisfy customer requirement but still space is available for improving it further

- THREATS
  - ➢ These are the problems /defects in software which result into failure

Workbench

- Term derived from engineering set up of mass production.
- Testers workbench consist of tools, guidelines, standards , process used for conducting test and evaluating whether process applied are effective or not?
- Must have entry criteria , process for doing work and exit criteria

- **Example of Testers Workbench**
  - Workbench for creating test strategy
  - Workbench for creating test plan
  - Workbench for writing test scenario
  - Workbench for writing test case
  - Workbench for execution
  - Workbench for defect management
  - Workbench for regression testing

Workbench

- Workbench for system testing execution
- **Inputs to Testers workbench**
  - Inputs may be test scenario, test case, test plan, work product, documentation for work product
- **Do Process**
  - Software undergoes testing as per defined test plan and procedure
  - Must guide normal tester while doing testing
- **Check Process**
  - Evaluating test process to compare achievements as defined in testing objectives
- **Output**
  - Must be available as required in the form of test report and test log from the test process

Workbench

- **Standards and Tools**
  - Includes how to install application, which steps are followed for testing, how ro capture defects
  - Also include defect management tools, configuration tools etc
- **Rework**
  - If check process find that do process are not able to achieve objectives defined for them , it must follow route of rework

Important Features of Testing Process

- Testing is a Destructive Process , But it is Constructive Destruction.
- Testing Needs a Sadistic Approach with Consideration That There is a Defect
- If the Test Does Not Detect Defect Present in System , it is Unsuccessful test
- A Test That Detects Defect is Valuable Investment for Development As Well As Customer, It Helps in Improving a Product
- It is Risky to Develop Software and Not to Test it Before Delivery
- With High Pressure to Deliver Software As Quickly As Possible , Test Process Must Provide Maximum Value in Shortest timeframe
- Highest Payback Comes from Detecting Defect Early in Software Development Life Cycle and Preventing Defect Leakage From One Phase to Another
- Organization's Aim Must Be Defect Prevention Rather Than Finding and Fixing Defect

Misconception About Testing

- **Anyone Can Do Testing , No Special Skills Are Required For Testing**
  - Have an approach any one can be put in testing
  - Developers on bench or people asking for light work put in testing
  - Test planning , test case writing/ execution not possible with unskilled people
- **Tester Can Test Quality of Product at the End of Development Process**
  - System or acceptance testing is considered as qualification testing for development
  - Few test cases out of infinite set of possibilities are used
  - Sometimes defect remain hidden for entire software development life cycle
- **Defect Found in Testing Are Blamed on Developers**
  - Though two third of defects are due to wrong requirements , yet developers are blamed
- **Defect Found By Customers Are Blamed on Tester**
  - 100% test coverage is not possible
  - No defect found during testing does not indicate product is defect free

Principles of Testing

#1: Goal of testing is to find defects BEFORE customers find them

#2: Program testing can only show presence of bugs, never their absence

#3: Test early and often

#4: Understand the “why” and not just the “what” / “how”

#5: Test the tests first

#6: Tests develop immunity – discover new tests!

#7: Testing encompasses defect prevention

#8: Plan the automation initiatives

#9: Instill pride in testers

Salient Features of Good Testing

- **Capturing User Requirements**
  - Expressed as well as implied requirements represent foundation on which software is built
  - Intended requirements analyzed and documented by tester to write test scenario and test case for these requirements
- **Capturing User Needs**
  - May be different from user requirements
  - May include present and future requirements and other requirements which may include process requirements
- **Design Objectives**
  - States why particular approach is selected for building software
  - The selection process indicates the reasons and criteria framework used for testing
- **User Interfaces**
  - The ways in which user interact with the system
  - Includes screens and other ways of communication with system and report generated by system

Salient Features of Good Testing

- **Internal Structure**
  - Mainly guided by software designs and guidelines or standards used for designing and development
  - May be defined by development organization as well as customers
- **Execution of Code**
  - Testing is execution of work product to ensure it works as intended by customer
  - Execution only proves that application module work correctly as defined in requirements

Test Policy, Strategy, Plan

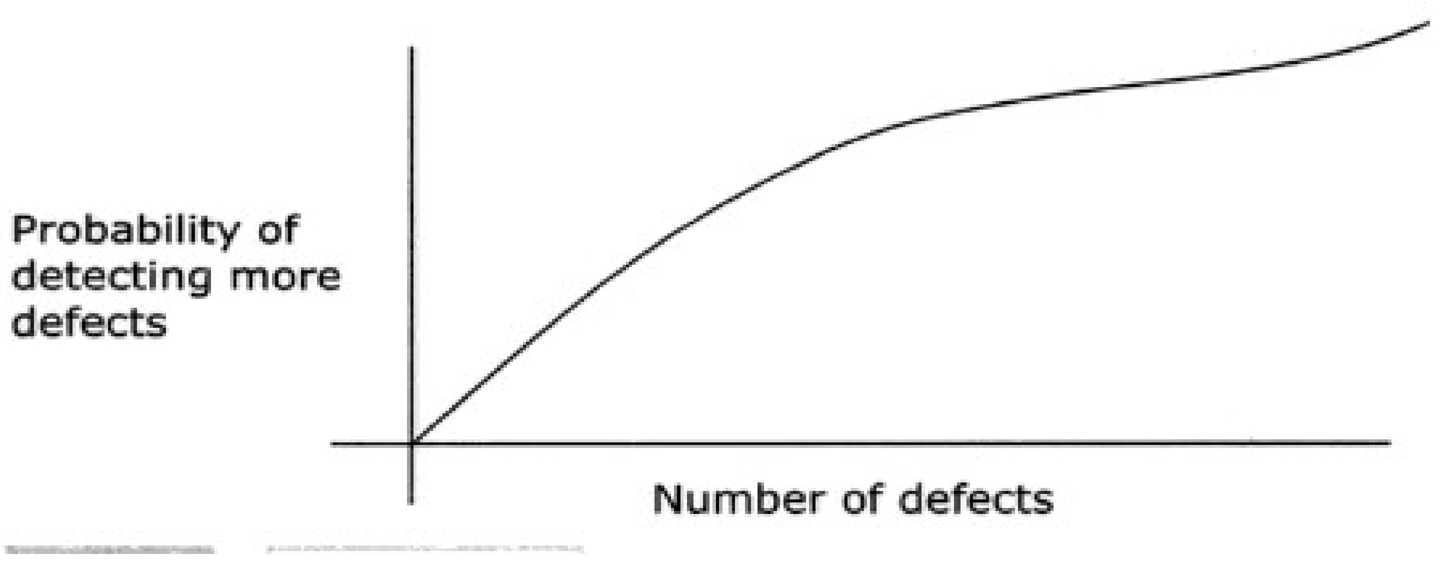
- **Test Policy**
  - Defined by senior management
  - Covers all aspects of testing
  - Provides framework for testing and its status in mission of achieving customer satisfaction
- **Test Strategy**
  - Defines action part of test policy
  - Provide ways and means of achieving policy
  - Differ from customer to customer, product to product ,time to time.
  - Includes Definition of coverage
  - Levels of testing
  - Manual and automation testing

Test Policy, Strategy, Plan

- **Test Plan**
  - First activity of test team
  - Intended to plan for testing throughout testing
  - Must be realistic and talks about limitations and constraints of testing
- **Points to remember**
  - Plan testing efforts adequately with an assumption that defects are there
  - If defects are not found ,it is failure of testing activity
  - Successful tester is not one who appreciates development but one who finds defect in the product
  - Testing is not formality to be completed at the end of development cycle

Testing Process and Number of Defects Found in Testing

- Testing is intended to find more number of defects .
- Finding more and more defects increases probability of finding more defects
- Base on principle every product has defect and testing team has efficiency to find more defects
- Governed by test team`s defect finding ability



Test Team Efficiency

- Very important aspect for development team and management
- If test team efficient in finding defects reduces number of testing iterations
- Test manager should aware about test team efficiency
- Ideally test team efficiency should be 100%, but in reality it is difficult to have it
- Suppose
  - Defects introduced by development team: X
  - Total defects found by Testing Team:Y
  - Defects found by testing team but not belonging to defects introduced by development team:Z
  - The ratio : (Y-Z)/X Gives test team efficiency

Mutation Testing

- It is used to check capability of test program and test case to find defects
- Faults are introduced into the program by creating many versions of the program called mutants.
- Each mutant contains a single fault.
- Test cases are applied to the original program and to the mutant program.
- The goal is to cause the mutant program to fail, thus demonstrating the effectiveness of the test case.
- Mutation testing involves the creation of a set of mutant programs of the program being tested.
- Suppose
  - Defects introduced by development :X
  - Defects found by test team in original program:Y
  - Defects found by test cases in mutatnt:Z
- Then
  - (Z-Y)/X gives test case efficiency

Mutation Testing

- Theoretically test team efficiency should be 100%
- But it may not be exactly 100% because of following reasons
- **Camouflage(hide) Effect**
- One defect Camouflage another defect and tester may not be able to see that defect and test case may not be able to locate that hidden defect
- **Cascading Effect**
- Due to existence of certain defects few more defects seen by tester
- There is no problem in modification defects are seen due to cascading effect
- **Coverage effect**
- 100% coverage is not possible
- Few lines of code or combinations are not tested and defect is introduced in such part
- **Redundant Effect**
- Part of code is not get executed under any conditions which is impossible to occur.
- If defect is present in such part then tester is not able to find these defects

Challenges in Testing

- Testing is very challenging job
- Challenges are different on different front
- Major challenges are
  - Requirements are not clear, complete, consistent, measurable, testable. These create problems in defining test scenario and test case
  - Requirements may be wrongly documented and interpreted by analyst
  - Code logic is difficult to capture
  - Error handling may be difficult to capture
  - Badly written code introduces many defects
  - Tester is always in lose -lose situation

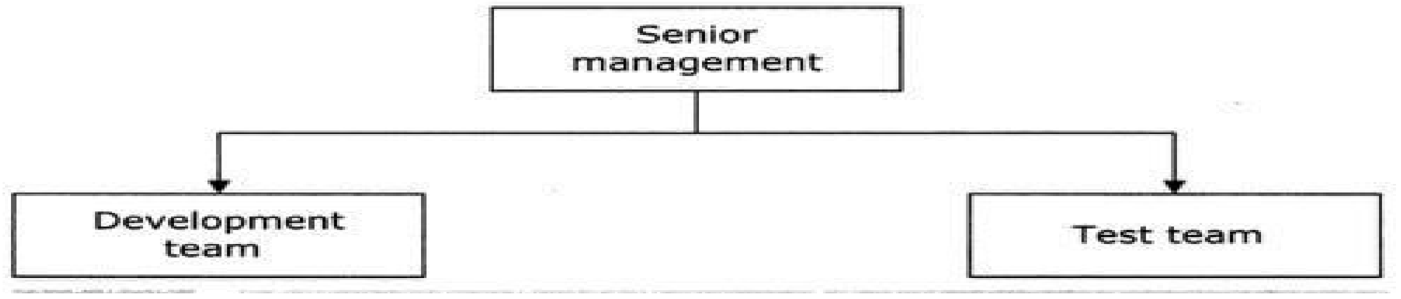
Test Team Approach

- Type of organization and type of product being developed define a test team

- **Location of Test Team in an Organization**
- Generally located in an organization as per testing policy
- Differs from organization to organization

Test Team Approach

1. **Independent Test Team**



Advantages:

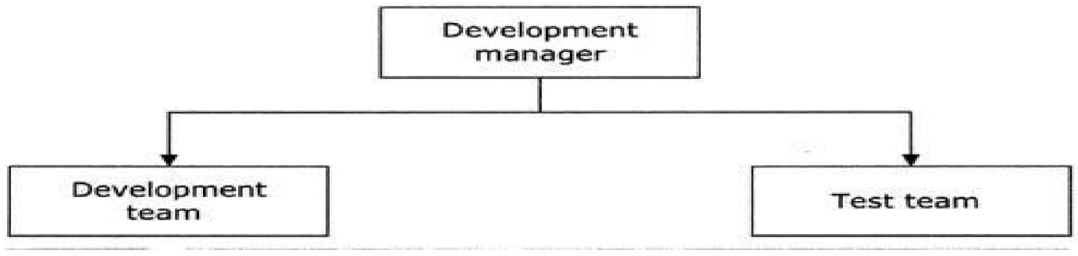
• Not under delivery pressure
• Not under pressure of ‘not finding defect’
• Independent view about product is obtained

Disadvantages

• There is always ‘us’ vs ‘then’ mentality between development team and test team
• Testers may not get complete understanding of product
• Management team may be inclined towards development or testing team

Test Team Approach

2. Test Team Reporting to Development Manager



Advantages:

• Better cooperation between development team and test team
• Test team involved in development and verification/validation activities from the start of project
• Tester may get good understanding of development process

• **Disadvantages**
• Defects found by testers considered as hurdles
• Tester test application from developers perspective which reduces their testing ability

Test Team Approach

- **Developers Becoming Testers**
- Developers in initial stage of SDLC take role of testers when latter stages of life cycle are executed

Advantages:

- No need of knowledge transfer while working as tester
- Developers have better understanding of coding and detail design
- Less costly
- Psychological acceptance of defects is not major problem

- **Disadvantages**
- Developers may not find values in testing
- Concentrates more on development activities like code creation and code optimization

Test Team Approach

- **Independent Testing Team**

Creates separate team with independent responsibility of testing

Advantages:

Concentrate more on test planning, test strategy and approach

Independent view of work product derived from requirement

- Testers working for customers can be seen in such scenario

- **Disadvantages**
- Additional cost for organization
- Need to check rivalries between development and testing team

Test Team Approach

- **Domain Experts Doing testing**

Employ domain experts for doing testing

Advantages:
- Fitness for use can be tested in this approach
- Domain experts testing is realistic

- **Disadvantages**
- It is difficult to get domain experts in diverse areas
- More costly

Process Problem Faced By Testing

- Defects in the product are due to incorrect process
- Incorrect process cause majority of problem
- The basic constitute of process are:

People

- ➢Many people are involved such as customer, analyst, developer tester

Material

- ➢Testers needs requirement document, development standards and test standards which add to their knowledge about product
- ➢These materials may not be available or may not be clear and complete

Machines

- ➢Testers build real life scenario using various machines, simulators
- ➢May include computers, hardware, software, printers

Methods

- ➢Methods for doing test planning, risk analysis, defining test scenario and test cases may not be proper

Cost Aspect of Testing

- Lets us consider project duration is = 10 months, 22 working days per month, 8 hours working per day, 100 people are working on it.
- Conversion rate is 500/- Per person per hour
- Cost of development and testing as follows
- Total efforts spent on project = 10*22*8*100 = 176,000 Hours
- **Total Cost = 176,000 * 500 = 88,000,000/-**
- Addition to this cost in terms of contingencies, overheads, and profit
- If contingency is 10%, overhead is 10%, expected profit is 20%
- **Sales price = 88,000,000 * 110% * 110% * 120%= 127,776,000**

Cost Aspect of Testing

- **Assessment of Cost of Testing**
  - Type of application and cost of testing
  - Development and test methodology
  - Domain and Technology aspect
  - Maturity of Development and testing processes

- **Cost of Prevention in Testing**
- **Cost of Appraisal in Testing**
- **Cost of failure in Testing**

Structured Approach To Testing

- Testing that is concentrated to single phase at the end of development life cycle is costly
- Can not show development process problem and same defect occur again and again
- Four components of waste involved in this type of testing

1. Waste in wrong development

2.Waste in Testing to detect defects

3. Wastage as wrong specifications, Design Codes and document must be replaced by Correct specifications, design, codes and documents

4.Wastage as system, must be retested to ensure that the corrections are correct

Categories of Defect

- Must be categorized under different criteria.
- Must be defined in test plan
- **Types of defects**
    - **Wrong Defect**
    - **Missing Defect**
    - **Extra**
- **Root causes of Defect**
    - Wrong requirement given by customer
    - Wrong interpretation of requirements
    - Wrong System architecture
    - Incorrect program specifications, guidelines, standards
    - Errors in code and errors in testing

Defect, Error and Mistake

•

Developing Test Strategy

- Test planning includes developing a strategy about how test team will perform testing
- Test strategy is a guideline to be followed to achieve the test objective and execution of test types mentioned in the testing plan.
  - **Step#1: Scope**
  - **Step#2 Test Approach/Levels**
  - **Step#3 Test Environment**
  - **Step#4 Testing Tools**
  - **Step#5 Release Control**
  - **Step#6 Risk Analysis**
  - **Step#7 Review and Approvals**

Developing Test Plan

- Acquire and study test strategy as defined earlier
- Determine the type of development project being executed
- Identify tactical risks
    - Structural Risks (Refers to methods used to build the product)
    - Technical Risks (Refers to technology used to build the product)
    - Size Risks (Refers to size of all aspects of software)
- Determine when testing must occurs during life cycle
- Type of development methodology

Testing Process

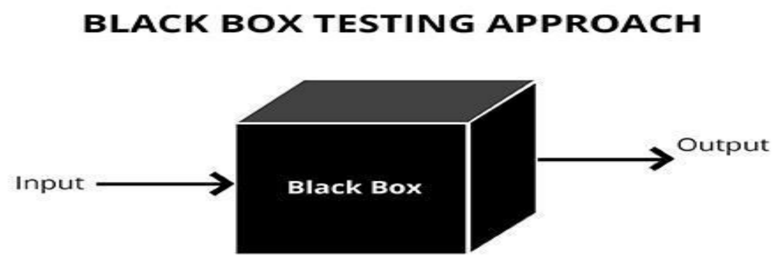
- It is made up of many milestones
- Testers need to achieve them , one by one, to achieve the final goal of testing
- Few milestones are
- **Defining test policy**
- **Defining test strategy**
- **Preparing test plan**
- **Establishing testing objectives to be achieved**
- **Designing test scenarios and test cases**
- **Writing/reviewing test cases**
- **Defining test data**
- **Creation of test bed**
- **Executing test cases**
- **Test Result**

Test Methodologies/ Approaches

- There are two main methods for doing testing
    - **BLACK BOX Testing**
    - Product is tested as per software specifications or requirement statement defined by business analyst
  - **WHITE BOX Testing**
    - Software is tested for structures
    - Covers verification of work product as per structure, architecture, coding standards
- One more method is there covering both methods at same time
  - **GREY BOX Testing**
  - Talks about combination of both approaches viz. black box and white box

Test Methodologies/ Approaches

- **BLACK BOX TESTING**



•**Advantages:**

1.Unbiased tests because the designer and tester work independently

2.Facilitates identification of contradictions and vagueness in functional specifications

3.Test is performed from a user’s point-of-view and not of the designer’s

4.Test cases can be designed immediately after the completion of specifications

•**Limitations**

5.Tests can be redundant if already run by the software designer

6.Test cases are extremely difficult to be designed without clear and concise specifications

7.Testing every possible input stream is not possible because it is time-consuming and this would eventually leave many program paths untested

Test Methodologies/ Approaches

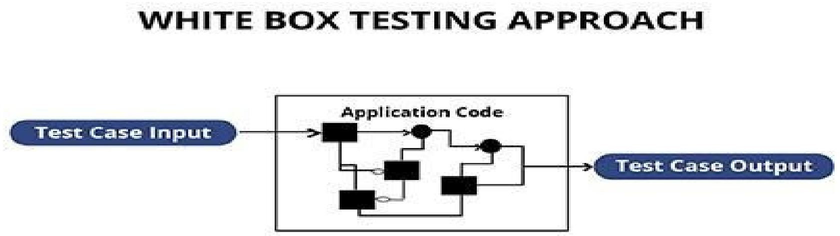
- **BLACK BOX TESTING**

- **Test Case Designing**

  1. Equivalence Partitioning
  2. Boundary Value Analysis
  3. Cause and Effect Graph
  4. State transition testing
  5. Use Case Based Testing
  6. Error Guessing

Test Methodologies/ Approaches

- **WHITE BOX TESTING**



• **Advantages:**

1. White box testing is primary method of verification

2. Only white box testing can ensures that developed procedures, methods of development have really been followed during software testing

3. White box testing or verification can gives early warnings, if something is not done properly.

4.Some characteristics of Software work product can be verified only. No chance of validating them. Ex. Code complexity and reuse of code

• **Limitations**

5.Does not ensures that users requirements are met correctly

6. It does not establish whether decision, condition path, and statements covered during review are sufficient or not for given set of requirements

7. Sometimes, white box testing is dominated by the usage of checklists.

Test Methodologies/ Approaches

- **WHITE BOX TESTING**

- **Test Case Designing**

1. Statement Coverage
2. Decision Coverage
3. Condition Coverage
4. Path Coverage
5. Logic Coverage

Test Methodologies/ Approaches

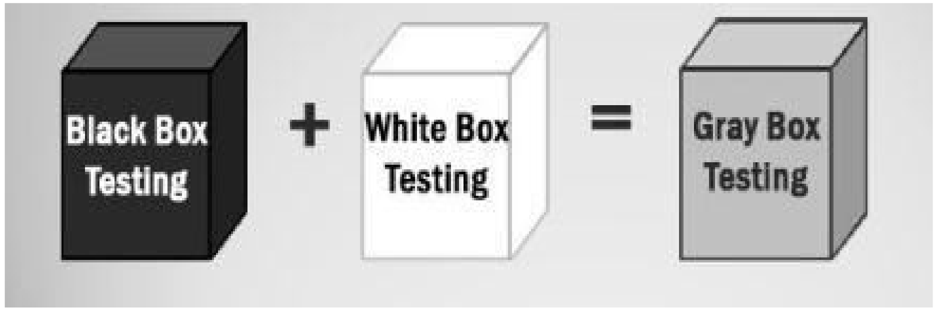
- **GREY BOX TESTING**

•**Advantages:**

1.Grey box testing offers combined benefit of both White box testing as well as Black box testing.
2.The testing will be performed from the user point of view instead of designer.
3.Testing is more thorough, with the possibility of covering most paths.

•**Limitations**

4.The ability to go over the code and test coverage is limited. Since the access to source code is not available.
5.Generally conducted with some automation tool



Skills Required by Tester

- Testing needs disciplined approach.
  - Testers working for client, finding obvious defects in processes and procedures

- **Skills are of two types**
  - General Skills
  - Testing Skills

- **General Skills:**
  - Written and Verbal Presentation Skills
  - Effective Listening Skills
  - Facilitation Skills
  - Software Development, Operations and maintenance
  - Continuous Education

Skills Required by Tester

- Testing Skills
- Concepts of Testing
- Levels of Testing
- Techniques for Verification and Validation
- Selection and Use of Testing Tools
- Knowledge of Testing Standards
- Risk Assessment and Management
- Developing Test Plan
- Defining Acceptance Criteria
- Checking of Testing Process
- Execution of Test Plan