Department of ICT

System Analysis and Design (SYAD-2432)

CHAPTER 6: System Testing and Quality Assurance

2nd Lut. Lencho A.

Masters in Computer Science and Engineering

E-Mail: **lenchoajema@gmail.com**

April, 2025

CHAPTER 6: System Testing and Quality Assurance

- Introduction
- Types of Testing
- Rules for System Testing
- Quality Assurance

Introduction

- - "Testing is the process of establishing confidence that a program or system does what it is supposed to." - Hetzel 1973
- - "Testing is the process of executing a program or system with the intent of finding errors." - Myers 1979
- - "Testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results." - Hetzel 1983

Introduction(continued):

- System testing is the process of evaluating a software system to ensure that it meets its requirements and works correctly in all possible scenarios.
- Quality assurance (QA) is a process that is used to ensure that the quality of a software system is maintained throughout the development process.

Importance of Software Testing:

- Last quality checking point for software on its production line
- Ensures the delivery of high-quality products or software applications
- Provides accurate, consistent, and reliable results
- Ensures effective performance of software applications
- Helps in gaining customer confidence and satisfaction
- Ensures business sustainability

When to Start and End Testing in SDLC:

- Testing should start as early as possible in the software development life cycle
- Testing should be focused on defined objectives
- Testing should continue until the required quality standards are met

Testing Limitations:

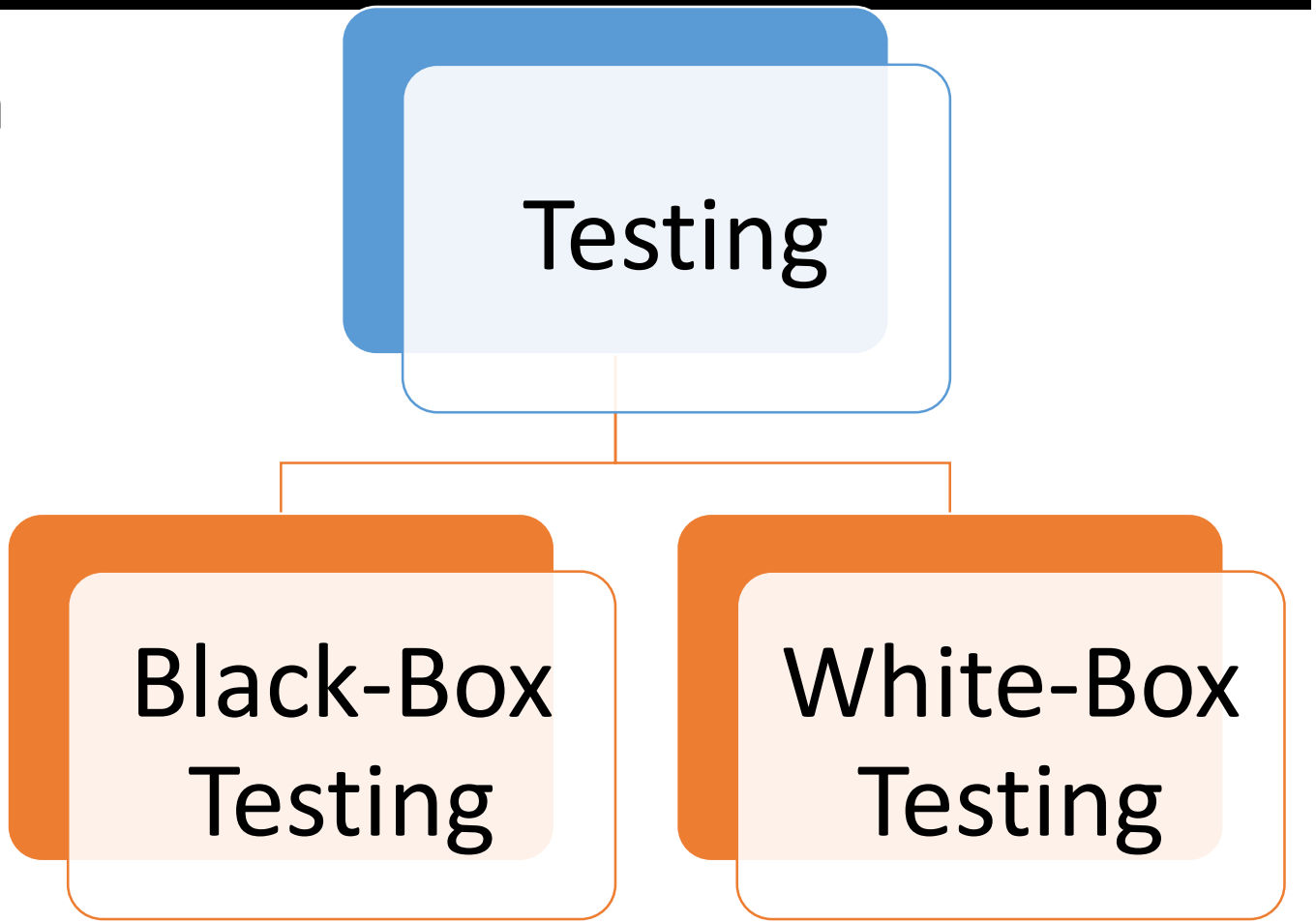
- 1) Testing shows the presence of defects but cannot prove the absence of defects
- 2) Exhaustive testing is impossible, so prioritization is necessary
- 3) Testing should start early in the development life cycle
- 4) Defect clustering suggests that a small number of modules contain most of the defects
- 5) The pesticide paradox implies that repetitive testing may become ineffective over time
- 6) Testing is context-dependent, varying based on different software types
- 7) Absence-of-errors fallacy - fixing defects does not help if the system is unusable

The Fundamental Test Process:

- 1) Test Planning and Control
- 2) Test Analysis and Design
- 3) Test Implementation and Execution
- 4) Evaluating Exit Criteria and Reporting
- 5) Test Closure Activities

Types of Testing

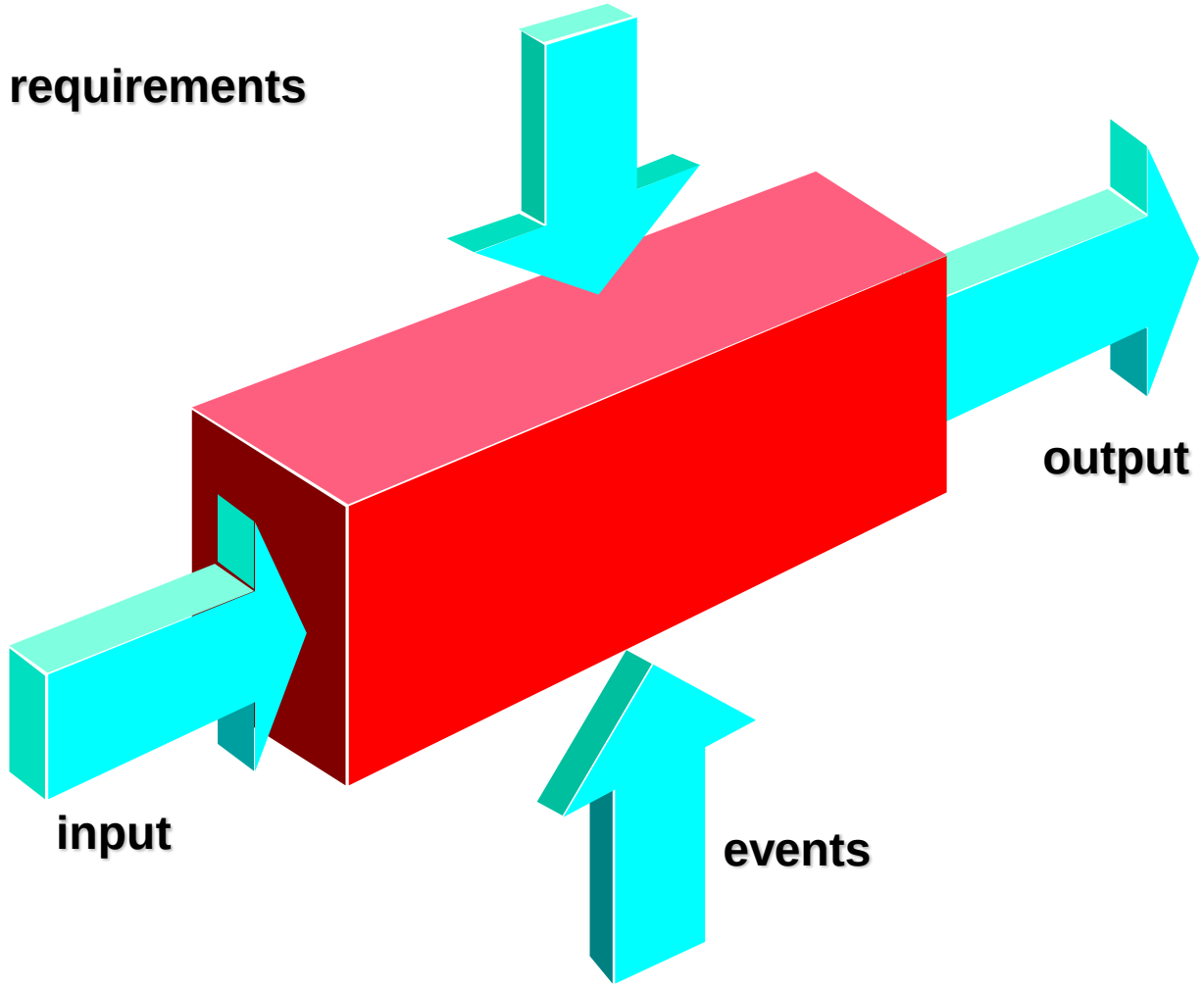
Testing is a crucial phase in software development that ensures the system functions correctly and meets the specified requirements.



Types of Testing:

- 1) Black Box Testing:
  - - Functional Testing
  - - Integration Testing
  - - System Testing
  - Acceptance Testing
  - - Regression Testing
  - - Performance Testing
  - - Security Testing
  - - Usability Testing
- 2) White Box Testing:
  - - Unit Testing
  - - Structural Testing

Black-Box Testing



Black-Box Testing

- Test cases are derived from formal specification of the system.
- Test case selection can be done without any reference to the program design or code.
- Only tests the functionality and features of the program.
  - Not the internal operation.

Black Box Testing:

- - Functional Testing: Testing the functionality of the software against the requirements
- - Integration Testing: Testing the interaction between different components/modules
- - System Testing: Testing the entire system as a whole

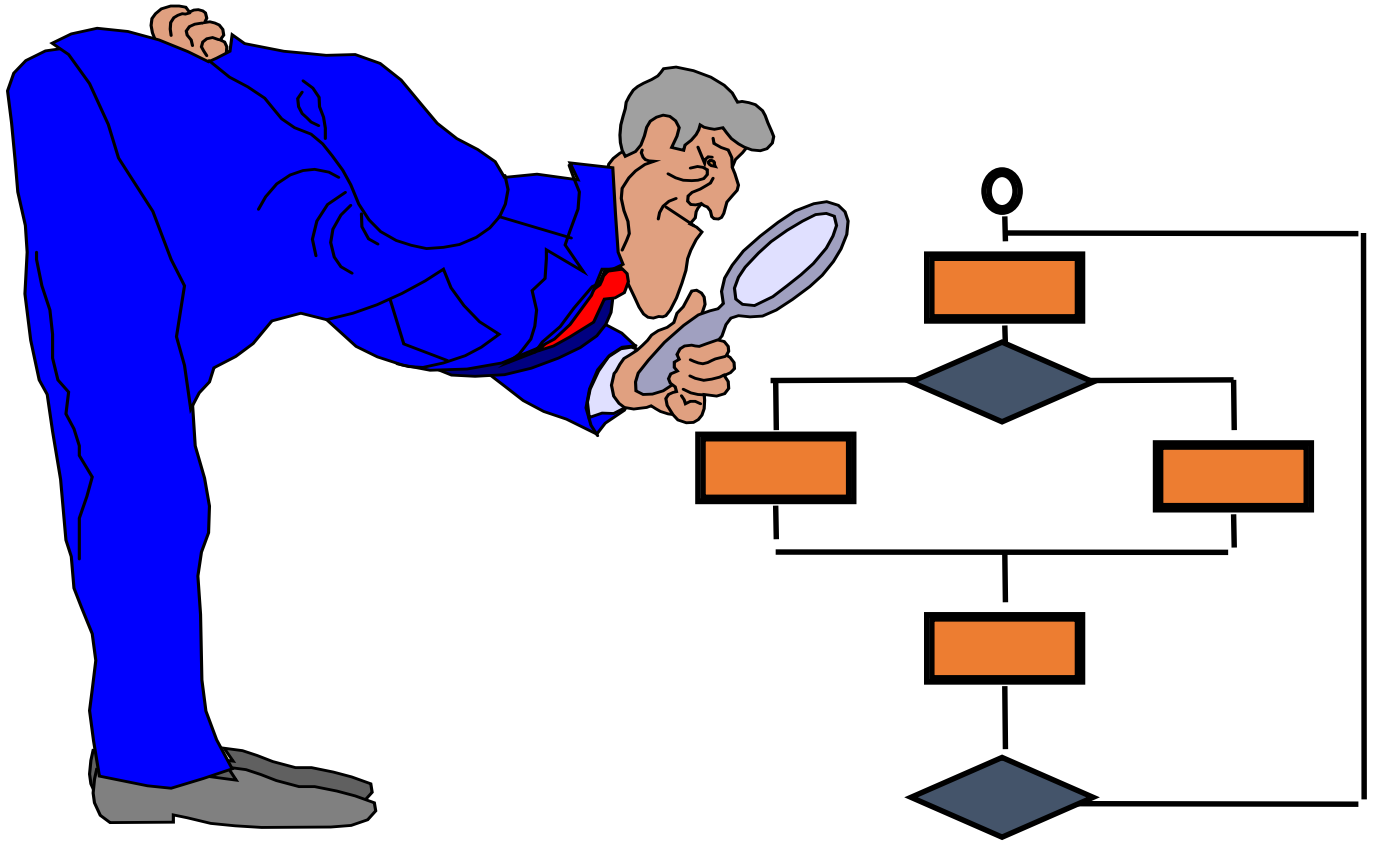
Black Box Testing (continued):

- Acceptance Testing: Testing the software from the end-user's perspective to determine if it meets the acceptance criteria
- Regression Testing: Testing to ensure that changes or modifications in the software do not introduce new defects or affect existing functionalities
- Performance Testing: Testing the performance and scalability of the software under different load conditions
- Security Testing: Testing the software for vulnerabilities and ensuring its resistance to attacks
- Usability Testing: Testing the software's user-friendliness and ease of use

Black-Box Testing

- Advantages
  - Test case selection is done before the implementation of a program.
  - Help in getting the design and coding correct with respect to the specification.

White-Box Testing



White-Box Testing

- Test cases are derived from the internal design specification or actual code for the program.
- Advantages
  - Tests the internal details of the code;
  - Checks all paths that a program can execute.
- Limitations
  - Wait until after designing and coding the program under test in order to select test cases.

White Box Testing:

- - Unit Testing: Testing individual components or units of code
- - Structural Testing: Testing the internal structure and logic of the software

Testing Techniques:

- Equivalence Partitioning: Dividing input values into equivalent classes to reduce the number of test cases
- Boundary Value Analysis: Testing values at the boundaries of input domains
- Decision Table Testing: Testing using a table that captures different combinations of inputs and corresponding outputs
- State Transition Testing: Testing the software's behavior as it transitions between different states
- Error Guessing: Intuition-based testing by guessing possible errors and designing test cases around them
- Exploratory Testing: Simultaneous learning, test design, and test execution based on real-time analysis

Challenges in Software Testing:

- Time and Resource Constraints
- Lack of Test Coverage
- Changing Requirements
- Complex and Dynamic Systems
- Test Environment Setup
- Communication and Collaboration
- Test Data Management
- Defect Management and Tracking

Best Practices for Effective Software Testing:

- Early involvement in the development process
- Clear and well-defined test objectives
- Adequate test coverage
- Proper test environment setup
- Effective test data management
- Test case prioritization
- Continuous integration and automation
- Regular communication and collaboration among team members
- Comprehensive defect management and tracking

Rules for System Testing:

- During system testing, several rules or principles should be followed to ensure effective and thorough testing. Some important rules include:
- a) Test Coverage: Aim for maximum coverage by testing all critical functionalities, paths, and scenarios. Ensure that both positive and negative test cases are covered.
- b) Test Environment: Create a dedicated and controlled test environment that closely resembles the production environment. This helps identify and replicate real-world issues accurately.

- c) Test Data: Use realistic and representative test data that covers a wide range of scenarios. This includes both valid and invalid data inputs.
- d) Test Documentation: Maintain comprehensive documentation of test cases, procedures, and results. This ensures traceability and facilitates collaboration among team members.
- e) Defect Tracking: Establish a robust defect tracking and management system. Record and prioritize defects, assign responsibilities, and track their resolution status.
- f) Test Automation: Utilize automation tools and frameworks to streamline repetitive and time-consuming test activities. This improves efficiency and reduces human errors.

Quality Assurance:

- Quality assurance (QA) is a set of activities and processes designed to ensure that the software development lifecycle adheres to defined quality standards. It aims to prevent defects and improve overall product quality. Key aspects of quality assurance include:
- a) Standards and Guidelines: Establishing and enforcing coding standards, design guidelines, and best practices to maintain consistency and readability of the codebase.
- b) Reviews and Inspections: Conducting code reviews, peer reviews, and inspections to identify defects, improve code quality, and share knowledge among team members.
- c) Process Improvement: Continuously evaluating and refining the development processes to enhance efficiency, reduce risks, and deliver high-quality software.

- d) Test Planning and Management: Planning and managing the testing activities throughout the software development lifecycle, including test strategy, test plan creation, test execution, and defect tracking.
- e) Metrics and Measurements: Defining and tracking key performance indicators (KPIs) to assess the effectiveness of quality assurance processes, identify bottlenecks, and make data-driven improvements.
- f) Continuous Improvement: Emphasizing a culture of continuous improvement by learning from past experiences, analyzing root causes of defects, and implementing corrective actions.

Difference between Testing, Quality Assurance, and Quality Control:

- Quality Assurance (QA): Focuses on implementing processes, procedures, and standards to ensure software verification.
- Quality Control (QC): Focuses on actual testing, executing software to identify bugs and defects.
- Testing: Subset of QC, focused on executing test cases to identify errors in the software.

Standards Related to QA and Testing Processes:

- IEEE 829 Test Documentation
- IEEE 830 Guide for developing system requirements specifications
- IEEE 730 Standard for software quality assurance plans
- IEEE 1008 Unit Testing
- IEEE 12207 Standard for software life cycle processes and life cycle data
- BS 7925-1 Vocabulary of Terms in Software Testing
- BS 7925-2 Software Component Testing Standard

Benefits of System Testing and Quality Assurance

- System testing and quality assurance can provide a number of benefits, including:
  - Increased confidence in the quality of the software system
  - Reduced risk of defects
  - Improved customer satisfaction
  - Increased productivity
  - Reduced costs

Conclusion:

- System testing and quality assurance are essential activities in the software development process.
- By following the rules of system testing and implementing a QA process, organizations can improve the quality of their software systems and reduce the risk of defects.
- It involves various types of testing techniques, both manual and automated.
- Test planning, execution, and evaluation are important phases of the testing process.
- Effective software testing requires collaboration, adherence to standards, and continuous improvement.