Testing Overview

Course Code: CSC4133 Course Title: Software Quality and Testing

Dept. of Computer Science
Faculty of Science and Technology

Lecturer No:	**8**	**Week No:**	**4**	**Semester:**	
Lecturer:	*Prof. Dr. kamruddin Nur,* *kamruddin@aiub.edu*				

Lecture Outline

- Software Testing: Concepts & Process
- Test Plan and Test case
- Seven Principles of Testing
- Testing Levels

Objectives and Outcomes

- **Objectives**: To understand the basic concept of testing and the testing process, to understand test plan and test case, to understand the principles of testing, to understand the different levels of testing.
- **Outcomes**: Students are expected to be able to explain the concept of testing and the testing process, be able to distinguish between test plan and test case, be able to explain the seven principles of testing, be able to explain the four levels of testing.

Software Testing

- *Software Testing is the process of executing a system or component under specified conditions with the intent of finding defects(bugs) and to verify that it satisfies specified requirements.*
- Testing is one of the most important parts of quality assurance (QA).
- Testing is the most commonly performed QA activity.
- Basic idea of testing involves the execution of software and the observation of its behavior or outcome.
- If a failure is observed, the execution record is analyzed to locate and fix the fault(s) that caused the failure. Otherwise, we gain some confidence that the software under test is more likely to fulfill its designated functions.

Seven Principles of Testing

- **Testing Principles:** A number of testing principles have been suggested over the past 40 years and offer general guidelines common for all testing.
- **Principle 1 – Testing shows presence of defects**
  - Testing can prove the presence of defects, but cannot prove the absence of defects. Even after testing the application or product thoroughly we cannot say that the product is 100% defect free. Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

Seven Principles of Testing

- **Principle 2 – Complete/Exhaustive testing is impossible**
  - Complete testing means testing a software with sorts of inputs (valid and invalid) under all execution environments.
  - For most of the systems, the input domain is huge, sometimes infinite.
  - Testing everything (all combinations of inputs and preconditions) is not feasible.
  - Instead of exhaustive testing, risk analysis, time & cost and priorities should be used to focus testing efforts.

Seven Principles of Testing

- **Principle 3 – Early testing**
  - To find defects early, testing activities should be started as early as possible in the software development life cycle, and should be focused on defined objectives.
  - When defects are found earlier in the lifecycle, they are much **easier** and **cheaper** to fix

Seven Principle of Testing

- **Principle 4 – Defect clustering**
  - A small number of modules usually contains most of the defects discovered during pre-release testing, or is responsible for most of the operational failures.
  - There is NO equal distribution of defects within one test object. The place where defect occurs, it's likely to find some more. The testing process must be flexible and respond to this behavior.

Seven Principle of Testing

- **Principle 5 – Pesticide paradox**
  - If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
  - To overcome this "pesticide paradox", test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to find potentially more defects.

Seven Principle of Testing

- **Principle 6 – Testing is context dependent**
  - Testing is done differently in different contexts.
  - For example, safety-critical software is tested differently from an e-commerce site.
  - All the developed software's are not identical. You might use a different approach, methodologies, techniques, and types of testing depending upon the application type.

Seven Principle of Testing

- **Principle 7 – Absence-of-errors fallacy**
  - Finding and fixing defects does not help if the system built is unusable and does not fulfill the users' needs and expectations.
  - Just because testing didn't find any defects in the software, it does not mean that the software is ready to be shipped.
  - It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement.

What is Test Plan?

- A **test plan** is a **document** that describes the objectives, scope, approach, resources, schedule and focus of software testing activities.
- A test plan gives detailed testing information regarding an upcoming testing effort. In other words, a test plan is a systematic approach to testing a system and typically contains a detailed understanding of what the eventual workflow will be.
- Organizations may follow some standard test plan outlines or they can have their own customized test plan outlines.

What is Test Case?

- A **test case** is a **document** that describes an input, action, or event, and its expected results, in order to determine if a feature of an application is working correctly.
- In other words, a test case is a document specifying inputs, predicted results and a set of execution conditions for a test item.
- Different organizations may use different test case formats.

What is the difference between Test Plan & Test Case?

- **Test Plan** is a **high-level document** whereas **Test Case** is a **low-level document.**
- A test plan is a comprehensive document that lays out all major activities associated with a particular testing project whereas a test case is only designed to test a particular scenario or feature.

What is a Test Scenario?

- A Test Scenario is defined as any functionality that can be tested. It is a collective set of test cases which helps the testing team to determine the positive and negative characteristics of the project.
- Test Scenario gives a high-level idea of what we need to test.
- Test Scenario is also called *Test Condition* or *Test Possibility*.

Examples of Test Scenario & Test Case

- **Examples of Test Scenario**
- For an e-Commerce Application, a few **test scenarios** would be
  - **Test Scenario 1:** Check the Search Functionality
  - **Test Scenario 2:** Check the Payments Functionality
  - **Test Scenario 3:** Check the Login Functionality

Examples of Test Scenario & Test Case

- **Examples of Test Cases**
- Test cases for the **Test Scenario: "Check the Login Functionality**" would be
  - Check system behavior when valid email id and password is entered.
  - Check system behavior when invalid email id and valid password is entered.
  - Check system behavior when valid email id and invalid password is entered.
  - Check system behavior when invalid email id and invalid password is entered.
  - Check system behavior when email id and password are left blank and Sign in entered.
  - Check Forgot your password is working as expected
  - Check system behavior when valid/invalid phone number and password is entered.
  - Check system behavior when "Keep me signed" is checked

What is a Test Suite?

- The collection of individual test cases that will be run in a test sequence until some stopping criteria are satisfied is called a test suite.
- Group of test cases that can be executed as a package in a particular sequence.
  - Test suites are usually related by the area of the system that they exercise, or by their priority, or by content.

Testing Levels

- Basically there are four levels of testing:
    1) Unit testing
    2) Integration testing
    3) System testing
    4) Acceptance testing

Books

1. *Software Testing and Quality Assurance: Theory and Practice*, by Kshirasagar Naik, Priyadarshi Tripathy
2. *Software Quality Assurance: From Theory to Implementation*, by Daniel Galin

References

1. *Software Quality Engineering: Testing, Quality Assurance and Quantifiable Improvement*, by Jeff Tian, published by Wiley, ISBN 0-471-71345-7, is the required text.
2. *Software Testing and Continuous Quality Improvement*, by William E. Lewis
3. *The Art of Software Testing*, by Glenford J. Myers, Corey Sandler and Tom Badgett
4. *Software Testing Fundamentals: Methods and Metrics* by Marnie L. Hutcheson