|  |  |  |
| --- | --- | --- |
| Related image | **KONERU LAKSHMAIAH EDUCATION FOUNDATION**  (Deemed to be University estd, u/s, 3 of the UGC Act, 1956) (NAAC Accredited “A++” Grade University)  Green Fields, Guntur District, A.P., India – 522502  **Department of Computer Science and Engineering**  (DST - FIST Sponsored Department) |  |

# B.Tech. II CSE(H) PROGRAM A.Y.2023-24 ODD, Semester-II 21CS3062RA

# Software Verification and validation

# CO1

# Session 1: TESTING LIFECYCLES, TEST CASES

1. **Course Description (Description about the subject)**

Software Verification and validation helps in finalizing the software application or product against business and user requirements. It is very important to have good test coverage in order to test the software application completely and make it sure that it’s performing well and as per the specifications

# Aim The primary objective of this course is to provide students with a comprehensive understanding of various techniques, methods, and tools used in the verification and validation of software. By studying this course, students will develop the necessary skills to assess and improve the quality of software through systematic testing, analysis, and verification techniques.

# Instructional Objectives (Course Objectives)

# Conducting validation activities: Students should learn how to plan and execute validation activities, including different types of testing such as functional testing, performance testing, usability testing, and security testing. They should understand how to design test cases, execute tests, and analyze the results to ensure the software meets user needs and performs as expected

# Learning Outcomes (Course Outcome)

# CO1: Illustrate fundamental concepts, terminologies, and techniques used in software verification and validation

# CO2: Identify different stages and activities involved in the software verification and validation process

# CO3: Apply the principles and theories underlying software verification and validation

# CO4: Analyze various software verification and validation artifacts, such as test plans, test cases, and defect reports

# Module Description (CO Description)

# Software verification and validation (V&V) involve fundamental concepts, terminologies, and techniques essential for ensuring software quality. Verification focuses on assessing compliance with requirements and specifications, while validation tests the software's real-world functionality. Techniques like black-box and white-box testing, unit testing, and system testing aid in uncovering defects and verifying system behavior. Key terminologies include test cases, plans, coverage, data, and scripts. Understanding these concepts and employing the right techniques enables effective V&V, resulting in high-quality and reliable software systems.

# Session Introduction

# This session will explore the fundamental aims of testing in software development. We'll dive into the specific objectives that drive testing efforts, such as validating functionality, uncovering defects, ensuring reliability, and evaluating performance. By understanding these objectives and goals, you'll gain insights into how testing contributes to delivering high-quality software. Let's embark on this journey to discover the purpose and significance of testing objectives and goals.

# Session description

# Testing objectives and goals refer to the specific aims and purposes of the testing process in software development. They define what testers aim to achieve through their testing activities and guide the overall testing strategy. Here is an elaboration on the concept of testing objectives and goals:

# Validate Software Functionality: One of the primary objectives of testing is to validate that the software functions correctly and performs the intended operations. Testers aim to ensure that the software meets the specified functional requirements and behaves as expected under different scenarios.

# Uncover Defects and Errors: Testing aims to identify defects, errors, or discrepancies in the software. Testers strive to discover issues that might cause the software to malfunction, produce incorrect results, or behave unexpectedly. By finding defects early in the development cycle, testing helps mitigate risks and facilitates their timely resolution.

# Ensure Software Reliability: Testing aims to assess the reliability of the software. Testers focus on ensuring that the software functions consistently and predictably, without unexpected crashes, errors, or failures. Reliability testing helps identify potential areas of weakness and improve the software's stability.

# Evaluate Performance: Testing objectives may include assessing the performance characteristics of the software. Testers measure and analyze factors such as response time, scalability, throughput, and resource utilization to ensure that the software meets the required performance standards and can handle expected user loads.

# Assess Software Usability: Usability testing aims to evaluate how user-friendly the software is and whether it meets the users' expectations. Testers focus on aspects such as ease of use, intuitive interfaces, clear instructions, and efficient workflows to ensure that the software provides a satisfactory user experience.

# Validate Security Measures: Testing objectives often encompass assessing the security of the software. Testers identify vulnerabilities, potential security breaches, or weaknesses in the system's security controls. By conducting security testing, testers help ensure that sensitive data is protected and the software adheres to security standards and regulations.

# Compliance with Standards and Regulations: Testing objectives may include verifying that the software complies with industry standards, regulations, or specific requirements. Testers ensure that the software meets relevant quality guidelines, legal requirements, accessibility standards, or other applicable regulations.

# Support Maintenance and Regression Testing: Testing aims to support future maintenance and regression testing efforts. Testers create test cases and scripts that can be reused to verify the software's behavior after changes, updates, or bug fixes. The objective is to ensure that modifications do not introduce new defects or regressions in existing functionality.

# Gain Confidence in the Software: Testing aims to instill confidence in the software's quality and reliability. By rigorously testing the software against various objectives, testers aim to provide stakeholders with assurance that the software is robust, performs as expected, and meets their needs.

# Optimize Testing Efforts: Testing objectives may include optimizing the testing process for efficiency and effectiveness. Testers aim to achieve maximum test coverage with a minimal number of tests, prioritize high-risk areas, and leverage test automation tools to streamline testing efforts.

# Testing objectives and goals are essential for setting clear expectations, guiding test planning and execution, and evaluating the success of the testing process. They help align testing activities with the project's objectives, stakeholder requirements, and overall quality goals. By defining and pursuing these objectives, testers can ensure that the software meets the desired quality standards and delivers value to end-users.

# Activities/ Case studies/related to the session 1) Defect Identification and Functional Testing in an E-commerce Website

# 2) Performance Testing and Scalability of a Banking Application

# 3) Usability Testing and User Experience Evaluation for a Mobile Banking App

# 4) Security Testing and Vulnerability Assessment of a Healthcare Software System

# 5) Compliance Testing for Regulatory Requirements in a Financial Software Application

# 6) Localization and Internationalization Testing for a Software Product

# 7) Regression Testing and Change Impact Analysis in a Software System

# 8) Reliability Testing and Fault Tolerance Evaluation of a Cloud-Based Storage System

# 9) Integration Testing and Interoperability Assessment of Software Components

# 10) User Acceptance Testing for Validating Business Requirements in a Software Release

1. **Examples & contemporary extracts of articles/ practices to convey the idea of the session**
2. Software Verification and Validation: Its Role in Computer Assurance and Its Relationship with Software Project Management Standards, “NIST Special Publication 500-165”
3. Validation, Verification, and Testing of Computer Software “NBS Special Publication 500-75”
4. <https://www.javatpoint.com/verification-and-validation-testing>
5. [www.geeksforgeeks.org](https://www.geeksforgeeks.org/software-engineering-verification-and-validation/)
6. https://www.guru99.com/verification-v-s-validation-in-a-software-testing.html

# SAQ's-Self Assessment Questions

# What is the purpose of software test metrics in the testing process?

# How do software test metrics help in measuring the effectiveness of the testing process?

# Can you explain the concept of defect density and its significance in software testing?

# How do test coverage metrics contribute to ensuring comprehensive testing?

# What are the challenges in effectively utilizing software test metrics?

1. **Summary**Software test metrics play a crucial role in evaluating the effectiveness and quality of the testing process. They provide objective measures that help assess various aspects, such as defect density, test coverage, and test case effectiveness. These metrics enable teams to track progress, identify areas for improvement, and make data-driven decisions. By utilizing software test metrics, organizations can enhance their testing strategies, mitigate risks, and continuously improve the overall quality of their software products.

# Terminal Questions Q1: Why are software test metrics important in the testing process?

# Q2: What are some commonly used software test metrics? Q3: How can software test metrics help in identifying areas for improvement in the testing process? Q4: What challenges might arise in the effective utilization of software test metrics?

1. **Case Studies (Co Wise)**1) Defect Identification and Functional Testing in an E-commerce Website

# 2) Performance Testing and Scalability of a Banking Application

# 3) Usability Testing and User Experience Evaluation for a Mobile Banking App

# 4) Security Testing and Vulnerability Assessment of a Healthcare Software System

# 5) Compliance Testing for Regulatory Requirements in a Financial Software Application

# 6) Localization and Internationalization Testing for a Software Product

# 7) Regression Testing and Change Impact Analysis in a Software System

# 8) Reliability Testing and Fault Tolerance Evaluation of a Cloud-Based Storage System

# 9) Integration Testing and Interoperability Assessment of Software Components

# 10) User Acceptance Testing for Validating Business Requirements in a Software Release

# Answer Key

# 

# Q1.Ans:

# Software test metrics are essential in the testing process for several reasons. Firstly, they provide objective and quantifiable measures of the quality and effectiveness of testing efforts. Metrics such as defect density, test coverage, and test case effectiveness offer insights into the level of code quality, thoroughness of testing, and the ability to detect defects. Secondly, software test metrics help in monitoring progress and tracking trends over time. By analyzing metrics, teams can identify areas of improvement, measure the impact of process changes, and make informed decisions to enhance the testing process. Overall, software test metrics enable organizations to assess the success of their testing efforts, drive continuous improvement, and deliver high-quality software products.

# Q2.Ans:

# There are several commonly used software test metrics that organizations utilize to evaluate the quality and effectiveness of their testing process. These metrics include:

# Defect Density: The number of defects identified per unit of code or per test case, indicating the level of code quality.

# Test Coverage: The percentage of code or requirements covered by executed test cases, assessing the comprehensiveness of testing.

# Test Case Effectiveness: The measure of how effective individual test cases are in detecting defects, helping identify inefficient or ineffective test cases.

# Mean Time to Failure (MTTF): The average time between the occurrence of a failure and its subsequent detection, providing insights into software stability and reliability.

# Defect Rejection Rate: The percentage of defects rejected or not accepted by the development team, indicating the accuracy and effectiveness of defect reporting and resolution processes.

# Test Execution Productivity: The measure of the efficiency of the testing team in executing test cases within a given time frame, assessing productivity and workload management.

# These metrics, among others, provide valuable information about the testing process, code quality, and the ability to detect and address defects effectively.

# Q3.Ans:

# Software test metrics play a crucial role in identifying areas for improvement in the testing process. By tracking and analyzing metrics, teams can uncover patterns, trends, and potential areas of concern. For example, if the defect density is consistently high in a particular module, it suggests the need for additional attention and focused testing in that area. Similarly, low test coverage in critical functionality indicates the need for expanding test scenarios and enhancing coverage. These metrics help teams pinpoint specific areas that require improvement and provide valuable insights for refining testing strategies, enhancing test coverage, optimizing test case effectiveness, and addressing process bottlenecks. By leveraging software test metrics, organizations can prioritize efforts, allocate resources effectively, and continuously enhance their testing process.

# Q4.Ans:

# While software test metrics offer valuable insights, there are challenges in their effective utilization. Firstly, selecting the appropriate metrics that align with project goals and objectives can be challenging. It requires understanding the specific needs and priorities of the project to choose the right metrics for measurement. Secondly, ensuring accurate data collection and reliable metrics can be a challenge. Teams need to establish robust data collection processes and ensure data accuracy and consistency. This may involve implementing automated tools, defining clear measurement criteria, and ensuring data integrity. Finally, interpreting and analyzing the metrics require domain expertise and a deep understanding of the software context. It is essential to avoid misinterpretation and draw meaningful insights from the metrics. Overcoming these challenges requires a combination of technical knowledge, process maturity, and effective collaboration among the testing team and stakeholders.

1. **Glossary  
     
   Defect Density**: The number of defects identified in a specific unit of code or per test case. It indicates the level of code quality and helps in evaluating the effectiveness of testing efforts.

**Test Coverage**: The measure of the extent to which the code or requirements are exercised by test cases. It helps assess the thoroughness of testing and identifies areas that require additional coverage.

**Test Case Effectiveness**: The measure of how well test cases detect defects. It evaluates the efficiency and quality of individual test cases in uncovering software issues.

**Mean Time to Failure (MTTF):** The average time between the occurrence of a failure and its detection during testing. It provides insights into the software's reliability and stability.

**Defect Rejection Rate:** The percentage of reported defects that are not accepted or rejected by the development team. It assesses the accuracy and effectiveness of the defect reporting process.

**Test Execution Productivity**: The measure of the efficiency and productivity of the testing team in executing test cases within a specific time frame.

**Code Complexity**: The measure of the complexity of the codebase, which affects the difficulty of testing and the likelihood of defects.

**Test Case Execution Progress**: The tracking of the progress of test case execution, including the number of executed, passed, failed, and pending test cases.

**Requirement Volatility**: The frequency and extent of changes to the software requirements. It helps assess the stability of requirements and its impact on testing efforts.

**Customer Satisfaction Index (CSI):** The measure of customer satisfaction with the quality and functionality of the software. It involves collecting feedback from end-users or stakeholders.

**Test Effort Variance:** The comparison between the actual effort expended on testing and the estimated effort. It helps assess the accuracy of effort estimation and resource allocation.

**Test Effectiveness Index**: The measure of the overall effectiveness of the testing process. It considers the number and severity of defects found during testing.

**Test Automation Coverage**: The percentage of test cases that are automated. It evaluates the level of test automation and its impact on testing efficiency.

**Test Environment Availability**: The measure of the readiness and availability of the test environment for executing test cases.

**Test Cycle Time**: The measure of the time taken to complete a testing cycle, from test planning to test closure**.**

# References of books, sites, links

# Text Books:

1. Ammann &Offutt, “Introduction To Software Testing”, Cambridge Univ Press

2. Naik,” Software Testing and Quality Assurance”, Wiley India pub.

3. Limaye,” Software Testing “, TMH Pub

**Reference Books:**

1. Paul C. Jorgensen, “Software Testing: A Craftsman's Approach, 3rd Edition, PHI 2014
2. Steven R. Rakitin, “Software Verification and Validation for Practitioners and Managers”, Second Edition
3. Aditya P. Mathur, Foundations of Software Testing \_ Fundamental Algorithms and Techniques 2ed Edition Pearson Education, (2008)

# Moocs links:

1. <https://www.coursera.org/learn/introduction-software-testing>
2. <https://www.classcentral.com/course/udacity-software-testing-365>

# 17. Keywords Test Case Execution Progress, Requirement Volatility, Customer Satisfaction Index (CSI), Test Effort Variance,Test Effectiveness Index