Date:27/11/24

Name: Venkatesh Kumar B

STLC

Defenition:

STLC stands for **Software Testing Life Cycle**. It is a series of phases or stages that are followed to ensure that a software application is thoroughly tested for quality and functionality before it is released.

Each phase in the STLC process focuses on different aspects of testing, from the initial planning and design to the final test execution and closure.

Requirement Analysis

Test Planning

Test Design

Test Environment Setup

Test Execution

Defect Reporting and Retesting

Test Closure

Requirement Analysis:

The first step involves understanding the software requirements and identifying the testable aspects. Testers collaborate with business analysts, stakeholders, and developers to gather the requirements and determine the scope of testing.

Purpose of Requirement Analysis in STLC:

Understanding Requirements:

The primary objective is to understand both functional and non-functional requirements of the system under test (SUT) in depth.

This includes user stories, business requirements, and technical specifications.

Identifying Testable Requirements:

Testers assess the requirements to identify which parts of the system can be tested. These may include user interface, functionalities, performance, security, and other areas.

Any ambiguous, unclear, or missing requirements are flagged for clarification.

Test Planning Input:

The requirement analysis phase serves as the input to the test planning phase. By analyzing the requirements, testers can begin to outline how testing will be conducted and what types of testing will be required (e.g., functional, performance, security, etc.).

Risk Identification:

This phase helps in identifying potential risks or areas that might require special attention, such as high-risk functionalities or critical business processes.

Clarifying Requirements:

The testing team works closely with stakeholders (e.g., business analysts, product owners, and developers) to clarify requirements and resolve any ambiguities.

This ensures that the testing process is based on accurate and well-defined information.

Deliverables of Requirement Analysis in STLC:

Requirement Traceability Matrix (RTM):

A critical deliverable that maps requirements to corresponding test cases. This helps ensure that all requirements are accounted for during testing and provides a way to track test coverage.

Test Planning:

In this phase, a test plan is created that outlines the strategy, objectives, resources, timelines, and deliverables.

It also includes risk management, roles, and responsibilities. The test plan acts as a roadmap for the entire testing process.

Purpose of Test Planning in STLC:

Define Testing Scope and Objectives:

The main objective of test planning is to clearly define what will be tested and what will not be tested, based on the requirements gathered in the previous phase.

This helps avoid ambiguity and ensures that all relevant functionality is covered.

Resource Allocation:

Planning the resources required for testing is a key purpose of the test planning phase.

This includes determining the team members, tools, testing environments, and hardware/software resources needed to execute tests effectively.

Set Testing Strategy and Approach:

The test planning phase defines the overall testing strategy, including which types of testing (e.g., functional, regression, performance, security) will be used, and how they will be executed.

It also includes the methodologies, such as manual vs. automated testing.

Establish Timeframes:

This phase helps set realistic timelines and milestones for the testing process, aligning the testing schedule with the overall project timeline

It ensures that enough time is allotted for each testing activity.

Risk Management and Mitigation:

The planning phase helps identify potential risks and uncertainties in the project.

A risk management plan is created to mitigate these risks and ensure that testing can proceed smoothly.

Define Deliverables:

Test planning ensures that all deliverables of the testing process, such as test cases, test data, defect reports, and test summary reports, are defined upfront.

This ensures that the team is aligned on what is expected at each stage.

Stakeholder Communication:

A detailed test plan ensures clear communication with stakeholders, such as developers, business analysts, and project managers.

It provides a common understanding of what will be tested, how, and when.

Test Design:

Test cases, test scenarios, and test scripts are created based on the requirements and test plan.

This phase focuses on preparing the actual tests that will be executed. Test data is also created or identified during this phase.

Purpose of Test Design in STLC:

Create Detailed Test Cases and Scenarios:

The primary goal of test design is to convert the high-level requirements and test strategy into detailed test cases, test scenarios, and scripts.

These should be clear, comprehensive, and directly linked to specific requirements to ensure adequate coverage.

Define Expected Results:

Each test case is designed with specific input data and expected results.

This is critical for validating the behavior of the application and for identifying discrepancies between actual and expected outcomes.

Test Coverage and Traceability:

The design phase ensures that every requirement, whether functional or non-functional, is tested.

Test cases are mapped back to requirements to ensure traceability, making it easy to verify that all requirements have been addressed.

Plan for Test Data:

Test data is an essential part of the design process.

The team identifies and prepares the data needed for testing, ensuring it covers a range of possible scenarios, including edge cases, boundary conditions, and invalid inputs.

Optimize Test Design for Efficiency:

During the test design phase, testers aim to create efficient, effective, and repeatable tests.

The design should balance thoroughness with efficiency to avoid unnecessary or redundant tests while ensuring high coverage.

Automation Readiness:

If test automation is part of the strategy, the test design phase may include identifying which tests are suitable for automation and preparing automated test scripts where necessary.

Test Environment Setup:

The necessary testing environment is set up, including hardware, software, network configurations, and any tools needed for testing.

This phase ensures that everything required to run tests is in place.

Purpose of Test Environment Setup in STLC:

Provide a Stable Testing Platform:
The primary purpose of test environment setup is to create a stable and consistent platform where tests can be executed.
The environment should mimic the production system as closely as possible to uncover issues that could potentially occur in the real-world deployment.

Replicate Production Environment:
Test environments should replicate the production environment, including the same operating systems, databases, servers, network configurations, and third-party integrations.
This ensures that tests run in a realistic setting, allowing the team to identify and fix defects early in the process.

Support Different Types of Testing:
Depending on the project, the test environment must support different types of testing (e.g., functional testing, load testing, integration testing, security testing).
The environment should be configured accordingly to allow for these various tests.

Ensure Reproducibility and Consistency:

Once the environment is set up, it should allow tests to be executed repeatedly with consistent results.
A stable test environment helps avoid variations caused by inconsistent configurations, ensuring that test results are reliable.

Support for Test Automation:

If automation is part of the testing strategy, the test environment must be configured to support automated testing tools and frameworks.
This includes setting up test automation tools, continuous integration (CI) servers, and ensuring compatibility with testing scripts.

Control and Isolation:
The test environment should be isolated from other environments (e.g., production or development) to prevent interference.
It should also provide controlled access to ensure that testers can safely execute tests without impacting other systems or users.

Test Execution:
This phase involves executing the test cases that were designed earlier.
Testers run the tests, report any defects or issues found, and track the results of each test.

Purpose of Test Execution in STLC:

Verify System Behavior:
The primary purpose of test execution is to verify that the application behaves as expected under various conditions.
This includes validating functionality, performance, usability, security, and other non-functional aspects, based on the predefined test cases.

Identify Defects and Issues:
During execution, testers identify defects (bugs or issues) by comparing actual outcomes to expected results.
The goal is to identify any discrepancies or failures that may impact the application's functionality or user experience.

Ensure Requirement Compliance:
Test execution helps ensure that the software meets the functional and non-functional requirements outlined in the earlier phases (requirements analysis, test planning, and test design).
Successful execution of tests provides confidence that the software meets these requirements.

Provide Feedback to Stakeholders:
Test execution provides valuable feedback to stakeholders (e.g., developers, product managers, business analysts) regarding the quality of the product.
It helps track the progress of testing and provides insights into the software's stability and readiness for release.

Track and Manage Test Progress:
The execution phase is key to tracking the overall testing progress, including the number of tests executed, passed, and failed.
Test execution also helps evaluate the effectiveness of the test cases and the efficiency of the testing process.

Defect Reporting and Retesting:
Any defects or issues found during test execution are reported to the development team for resolution. After defects are fixed, the affected test cases are rerun to verify the fixes (retesting).
If the fixes work as expected, the issues are marked as closed.

Defect Reporting in STLC: Purpose and Use:

Purpose of Defect Reporting:
Defect reporting is the process of identifying, documenting, and tracking defects (bugs) found during test execution.
When a test case fails or an issue is found, testers need to report the defect to the development team so that it can be investigated, reproduced, and resolved.

Identify and Log Issues:
The purpose is to systematically record any defects encountered during testing, ensuring that they are tracked, prioritized, and addressed by the development team.
Provide Detailed Information:

The defect report needs to provide enough information for developers to understand the problem, replicate it, and apply a fix.

It should include details such as the severity, environment, and steps to reproduce the issue.

Ensure Accountability and Traceability:

Defect reporting helps maintain traceability throughout the testing process.
It ensures that all defects are tracked, managed, and fixed in the right sequence, helping ensure that no critical issues are overlooked.

Facilitate Communication:

Defect reports act as communication tools between the testing and development teams.

Proper documentation helps developers prioritize the issues, and it provides a basis for further discussions regarding fixes or workarounds.

Test Closure:
After testing is complete, the test team documents the results, performs a final assessment of the testing process, and prepares reports for stakeholders.
The final outcome is reviewed, and testing is formally closed.

Purpose of Test Closure in STLC:

Conclude Testing Activities:

The primary purpose of test closure is to formally conclude all testing efforts, ensuring that all planned test cases have been executed and all defects have been logged and resolved, or appropriately deferred.

Assess Testing Coverage and Quality:
It helps assess the test coverage and overall quality of the product.

This includes evaluating whether all features and requirements have been tested and whether the software meets the desired quality standards before it is handed over for release.

Ensure All Test Deliverables are Completed:

This phase ensures that all testing deliverables are completed and delivered to stakeholders.
These may include test results, defect logs, test reports, and other documents that summarize the outcomes of the testing process.

Identify Process Improvements:
Test closure provides an opportunity to identify and document lessons learned during the testing process.
This can help improve the testing process for future projects, optimize test strategies, and highlight areas for improvement in tools, methodologies, or team coordination.

Create and Archive Test Artifacts:
Test artifacts, including test cases, test scripts, test logs, and defect reports, need to be archived for future reference, audits, or post-release verification.
This also ensures that all test documentation is available for any future regression or retesting efforts.

Prepare for Transition:
Test closure ensures a smooth transition from the testing phase to the release phase.
It involves ensuring that the product is ready for deployment, that all critical defects have been addressed, and that the stakeholders are satisfied with the quality of the product.