Software Testing Strategies – Types of Software Testing Strategies

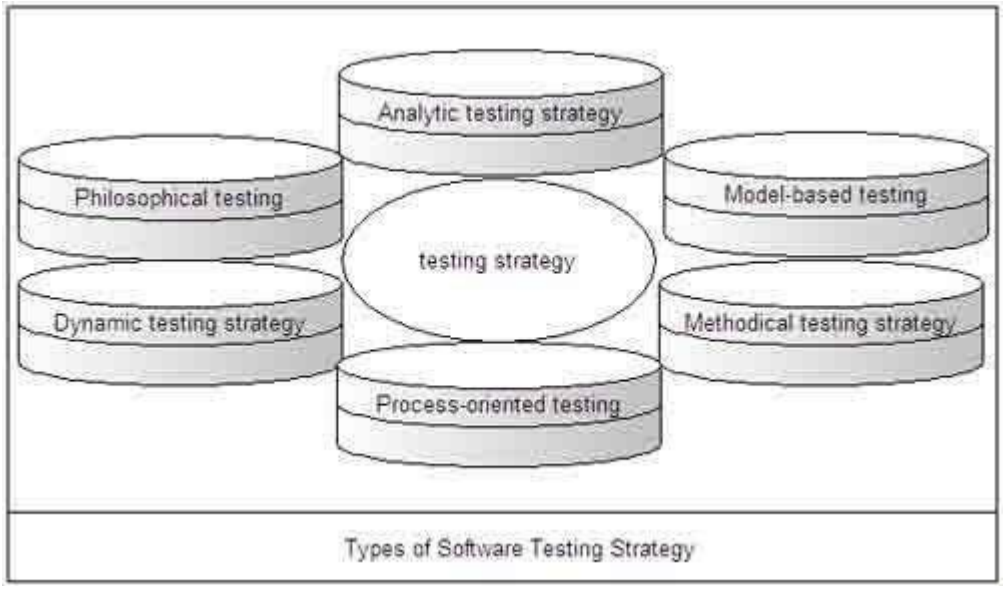
To perform testing in a planned and systematic manner, software testing strategy is developed. A testing strategy is used to identify the levels of testing which are to be applied along with the methods, techniques, and tools to be used during testing. This strategy also decides test cases, test specifications, test case decisions, and puts them together for execution.

Developing a test strategy, which efficiently meets the requirements of an organization, is critical to the success of software development in that organization.

The choice of software testing strategy is highly dependent on the nature of the developed software. For example, if the software is highly data intensive then a strategy that checks structures and values properly to ensure that all inputs given to the software are correct and complete should be developed. Similarly, if it is transaction intensive then the strategy should be such that it is able to check the flow of all the transactions. The design and architecture of the software are also useful in choosing testing strategy. A number of software testing strategies are

developed in the testing process. All these strategies provide the tester a template, which is used for testing. Generally, all testing strategies have following characteristics.

1. Testing proceeds in an outward manner. It starts from testing the individual units, progresses to integrating these units, and finally, moves to system testing.
2. Testing techniques used during different phases of software development are different.
3. Testing is conducted by the software developer and by an ITG.
4. Testing and debugging should not be used synonymously. However, any testing strategy must accommodate debugging with itself.



Types of Software Testing Strategies

There are different types of software testing strategies, which are selected by the testers depending upon the nature and size of the software. The commonly used software testing strategies are listed below.

- **Analytic testing strategy:** This uses formal and informal techniques to access and prioritize risks that arise during software testing. It takes a complete overview of requirements, design, and implementation of objects to determine the motive of testing.
- **Model-based testing strategy:** This strategy tests the functionality of the software according to the real world scenario (like software functioning in an organization). It recognizes the domain of data and selects suitable test cases according to the probability of errors in that domain.
- **Methodical testing strategy:** It tests the functions and status of software according to the checklist, which is based on user requirements. This strategy is also used to test the functionality, reliability, usability, and performance of the software.
- **Process-oriented testing strategy:** It tests the software according to already existing standards such as the IEEE standards. In addition, it

checks the functionality of the

software by using automated testing tools.

- **Dynamic testing strategy:** This tests the software after having a collective decision of the testing team. Along with testing, this strategy provides information about the software

  such as test cases used for testing the errors present in it.

- **Philosophical testing strategy:** It tests the software assuming that any component of the software can stop functioning anytime. It takes help from software developers, users and

  systems analysts to test the software.

A testing strategy should be developed with the intent to provide the most effective and efficient way of testing the software. While developing a testing strategy, some questions arise such as: when and what type of testing is to be done? What are the objectives of testing? Who is responsible for performing testing? What outputs are produced as a result of testing? The inputs that should be available while developing a testing strategy are listed below.

- Type of development project
- Complete information about the hardware and software components that are required to develop the software
- Risks involved
- Description of the resources that are required for testing
- Description of all testing methods that are required to test various phases of SDLC
- Details of all the attributes that the software is unable to provide. For example, software cannot describe its own limitations.

The output produced by the software testing strategy includes a detailed document, which indicates the entire test plan including all test cases used during the testing phase. A testing strategy also specifies a list of testing issues that need to be resolved.

An efficient software testing strategy includes two types of tests, namely, low-level tests and high-level tests. Low-level tests ensure correct implementation of small part of the source code and high-level tests ensure that major software functions are validated according to user requirements. A testing strategy sets certain milestones

for the software such as final date for completion of testing and the date of delivering the software. These milestones are important when there is limited time to meet the deadline.

In spite of these advantages, there are certain issues that need to be addressed for successful implementation of software testing strategy. These issues are discussed here.

- In addition to detecting errors, a good testing strategy should also assess portability and usability of the software.
- It should use quantifiable manner to specify software requirements such as outputs expected from software, test effectiveness, and mean time to failure which should be

  clearly stated in the test plan.
- It should improve testing method continuously to make it more effective.
- Test plans that support rapid cycle testing should be developed. The feedback from rapid cycle testing can be used to control the corresponding strategies.
- It should develop robust software, which is able to test itself using debugging techniques.
- It should conduct formal technical reviews to evaluate the test cases and test strategy. The formal technical reviews can detect errors and inconsistencies present in the testing

  process.

Characteristics of STLC

- STLC is a fundamental part of the SDLC but STLC consists of only the testing phases.
- STLC starts as soon as requirements are defined or software requirement document is shared by stakeholders.
- STLC yields a step-by-step process to ensure quality software.

Phases of STLC

1. Requirement Analysis: Requirement Analysis is the first step of the Software Testing Life Cycle (STLC). In this phase quality assurance team understands the requirements like what is to be tested. If anything is missing or not understandable then the quality assurance team meets with the stakeholders to better understand the detailed knowledge of requirements.

The activities that take place during the Requirement Analysis stage include:

- Reviewing the software requirements document (SRD) and other related documents
- Interviewing stakeholders to gather additional information
- Identifying any ambiguities or inconsistencies in the requirements
- Identifying any missing or incomplete requirements
- Identifying any potential risks or issues that may impact the testing process

Creating a requirement traceability matrix (RTM) to map requirements to test cases At the end of this stage, the testing team should have a clear understanding of the software requirements and should have identified any potential issues that may impact the testing process. This will help to ensure that the testing process is focused on the most important areas of the software and that the testing team is able to deliver high-quality results.

2. Test Planning: Test Planning is the most efficient phase of the software testing life cycle where all testing plans are defined. In this phase manager of the testing, team calculates the estimated effort and cost for the testing work. This phase gets started once the requirement- gathering phase is completed.

The activities that take place during the Test Planning stage include:

- Identifying the testing objectives and scope
- Developing a test strategy: selecting the testing methods and techniques that will be used
- Identifying the testing environment and resources needed
- Identifying the test cases that will be executed and the test data that will be used
- Estimating the time and cost required for testing
- Identifying the test deliverables and milestones
- Assigning roles and responsibilities to the testing team
- Reviewing and approving the test plan

At the end of this stage, the testing team should have a detailed plan for the testing activities that will be performed, and a clear understanding of the testing objectives, scope, and

deliverables. This will help to ensure that the testing process is well-organized and that the testing team is able to deliver high-quality results.

3. Test Case Development: The test case development phase gets started once the test planning phase is completed. In this phase testing team notes down the

detailed test cases. The testing team also prepares the required test data for the testing. When the test cases are prepared then they are reviewed by the quality assurance team.

The activities that take place during the Test Case Development stage include:

- Identifying the test cases that will be developed
- Writing test cases that are clear, concise, and easy to understand
- Creating test data and test scenarios that will be used in the test cases
- Identifying the expected results for each test case
- Reviewing and validating the test cases
- Updating the requirement traceability matrix (RTM) to map requirements to test cases