STQA

1.1 Goals (PDF Pg 19-21)

Short Term	Long Term	Post-implementation
OR Immediate	→ Quality	→ Reduced
→ Bug Discovery	↳ Reliability	Maintenance cost
→ Bug Prevention	→ Customer Satisfaction	→ Improved Testing Process,
	→ Risk Management	

Short Term: These goals are the immediate results after performing testing. Can be applied/set in the individual phases of SDLC

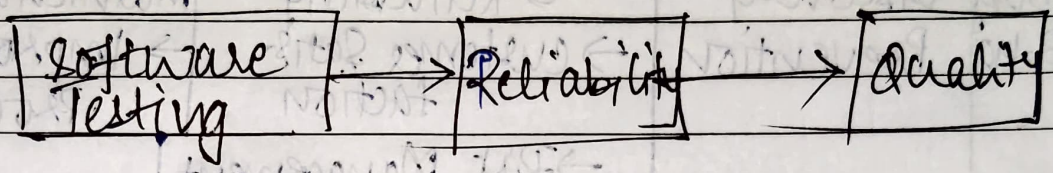
→ Bug Discovery: The main goal of testing is to find errors at any stage of software development. More the bugs discovered at an early stage, higher will be the success rate of software testing

→ Bug Prevention: Consequent action of bug discovery. After analyzing the behaviour and interpriting the discovered bugs the entire software development team learns to code safely so that the same error does not appear in later stages or future projects. Errors cannot be reduced to zero but can be minimized.

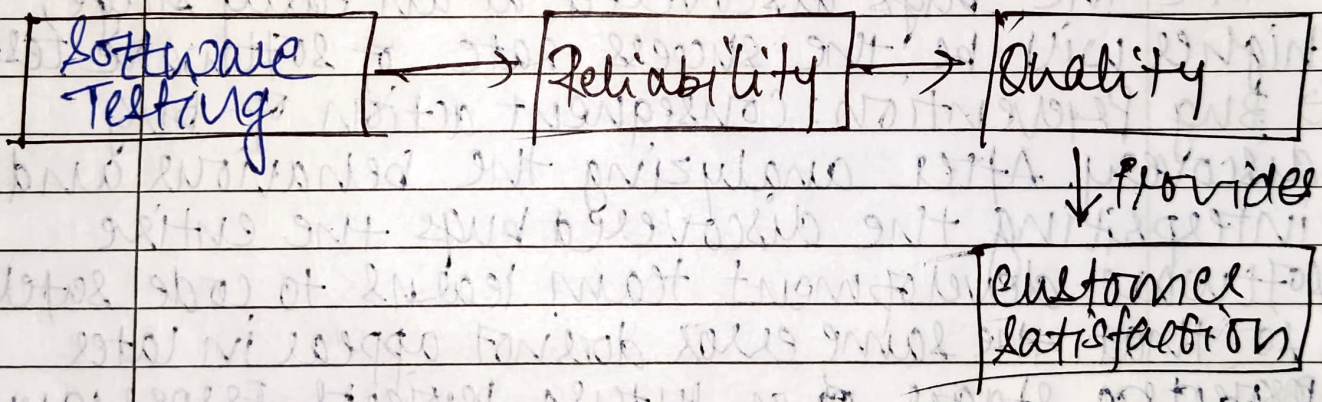
Long Term: Goals that affect the product quality in the long run when one cycle of SDLC is over.

Quality: Software is also a product and its quality is the primary factor for users. Thorough testing ensures good quality. First goal is to enhance the quality of the software product. Quality depends on various factors but the most important

is reliability. Reliability is a confidence factor that ensures the software will not fail. To achieve high quality standards the software needs to be passed through rigid/rigorous reliability analysis.

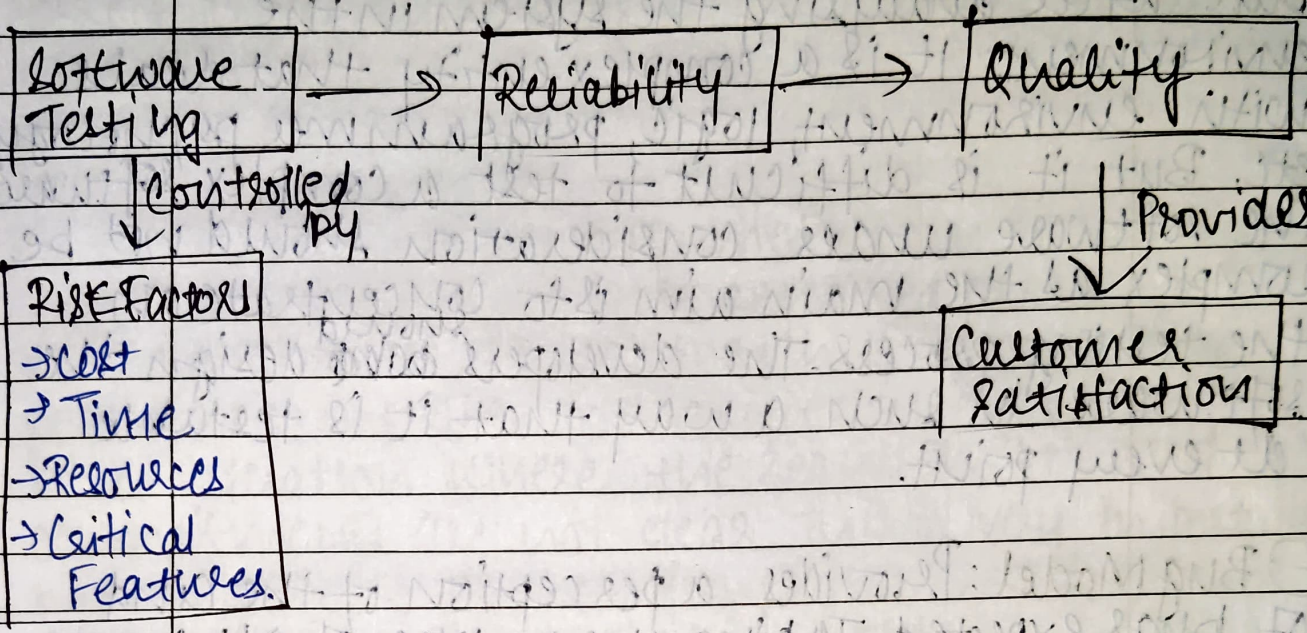


Customer Satisfaction: From users perspective the prime concern of testing is customer satisfaction. Testing should be thorough and complete. All the requirements mentioned by the customer in the user manual must be satisfied. Even the unspecified requirements should be understood.



Risk Management: Risk is the probability that an undesired event will occur in the system which will prevent the organization from successfully implementing its business initiatives. Risk should be managed in a way such that they can be controlled with ease. Software testing acts as a risk controller. Its main purpose is to provide information to the management so that they can better react to such situations.

Testers should also categorize the level of risk after assessment



Post Implementation Goals: These goals are important after the software is released.

Reduced Maintenance cost: The only maintenance cost in a software product is its failure due to errors. Post release errors are costlier to fix as they are difficult to detect. Thus...

Improved software testing: A testing process for one project may not be successful or there may be scope of improvement. Therefore bug history and post implementation results can be analyzed to find out snags in current testing process which can be rectified for future projects.

__/__/____

Models for Software Testing:

① Software and Software Model: Software is built after analysing the system in the environment. It is a complex entity that deals with environment, logic, programmer psychology etc. But it is difficult to test a complex software. The software under consideration should not be complex as the main aim is to concentrate on the testing process. The developers should design the software in such a way that it is testable at every point.

② Bug Model: Provides a perception of the kind of bugs expected. Taking the nature of all types of bugs in consideration a testing strategy is decided. Not every kind of bug can be predicted therefore, for every incorrect result the bug model needs to be modified.

③ Testing Methodology and Testing: Based on the input on the software model and bug model testers can develop a testing methodology that incorporates both testing strategies and tactics. Testing strategy is a road map that gives well-defined steps for the testing process. Once these steps are prepared software testing tools and techniques can be applied. However, if the desired results are not achieved the testing plan must be checked and modified accordingly.

__/__/____

1.2

Verification: A set of activities that ensures correct implementation of specific ~~[illegible]~~ functions in a software.

Need for Verification:

→ If verification is not performed at an early stage there is a chance that the required product and delived product do not match. If ~~[illegible]~~ requirements are not verified there might be a situation where the requirement and commitments are not clear. This is very important when it comes to non-functional requirements and increases the probability of bugs.
→ Early verification reduces the cost of fixing bugs.
→ Verification exposes more errors.
→ Enhances the quality of software.

Goals of Verification:

→ Everything must be verified: All SDLC phases and all the products of these phases must be verified.
→ Results of verification may not be binary.
→ Even implicit qualities must be verified.

Verification of Requirements: In this type of verification all the requirements gathered from the users viewpoint are verified. Requirement verification has high potential of detecting bugs. Testers use SRS for verification.

__/__/____

PDF

Points against which SRS should be verified: (Pg 87)

① Correctness:
→ No tools or procedure to measure correctness of a specification
→ Testers use their intelligence to verify the correctness.
→ Testers should refer to other documentation to compare the specified requirement.
→ Testers should reach out to the user if any requirement is not understood.
→ Tester should check the correctness in the sense of realistic requirement.

② Unambiguous:
→ A requirement should be verified such that it does not provide too many interpretations.
→ There should not be any redundancy.
→ Each characteristic should be described using a single term so that there are no bugs due to ambiguity.

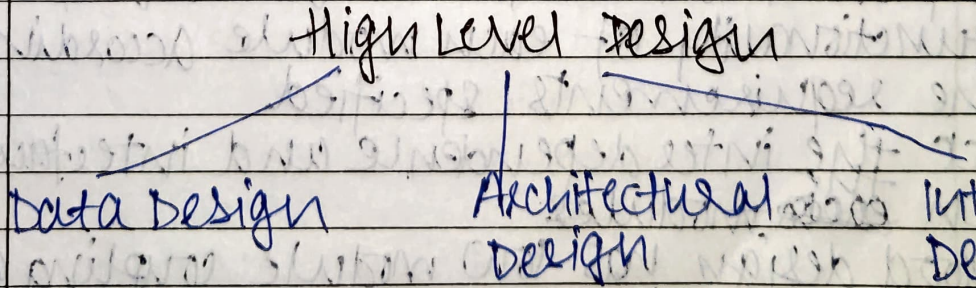
③ Consistent:
Specifications should not contradict or conflict with each other. Must be checked for
→ Real-world objects conflict
→ Logical conflict between two specified actions
→ Conflicts in terminology.

④ Completeness
→ Verify that all significant requirements such as functionality, performance etc are completed.

→ check whether responses of every possible input to the software is defined.
→ check whether all tables and figures are labeled and referenced completely.

5. Updation
→ If the specification is a newone, all the above mentioned steps and their feasability must be verified.
→ For any changes, the change should be verified that it can be implemented in the current design.

6. Traceability
The traceability of requirements must also be verified such that the origin of each requirement is clear. The following 2 types of traceability must be verified.
→ Backward Traceability.
→ Forward Traceability



__/__/____

Verification of Data Design:

→ Check whether the sizes of data structure have been estimated approximately
→ Check the provisions of overflow in a data structure
→ Check the consistency of data formats with the requirements
→ Check whether data usage is consistent with it declaration
→ Check relationships among data objects.
→ Check consistency of database and data warehouses with the requirement specified in the SRS.

Verification of Architectural Design.

→ Check that every functional requirement has been taken care of in the design.
→ Check if all exceptional handling conditions have been taken care of.
→ Verify the process of transform mapping and transaction mapping used for the transition from requirement model to architectural design
→ Architectural design deals with the classification of a system into subsystems or modules, check the functionality of each module according to the requirements specified.
→ Check the inter dependence and interface between modules
→ A good design has low module coupling and high module cohesion. Testers should verify these factors or else they will affect the reliability and maintainability of the system.

__/__/____

Verification of User Interface Design

→ Check all interfaces between modules
→ Check all interfaces between software and other non-human producer and consumer of info.
→ Check all interfaces between human and computer.
→ Check all the above interfaces for consistency.
→ Check if the response time for all the interfaces are within required range especially for real time projects.
→ For Help facility verify:
↳ The representation of Help in it desired manner
↳ The user returns to normal interaction from Help.
→ For error messages and warnings verify:
↳ Whether the message clarifies the problem.
↳ Whether the message provides constructive advice for recovering from the error.
→ For typed command interaction, check the mapping between every menu option and their corresponding commands.

__/__/____

Unit Testing: The strategy is to first focus on the building blocks of the full system. A unit or module is the basic building block of the full system that can be tested for its functionality and interfaces. Thus, unit testing is a process of testing the individual components of a system. A module/unit must be validated before integrating it with other modules. It is the first validation activity after the coding of one module is over.

Motivation:

→ If the whole software is tested at once it becomes difficult to trace the bug.

→ Since the developer has his attention focused on a module, it is quite natural to test the unit first.

→ In large scale projects there might be a number of modules present and each module may contain tens of thousands of lines of code. It is not practically possible for a single developer to develop all modules. Many other developers are involved here. This requires parallelism in software development.

Integration Testing: Process of combining and testing multiple modules together. The individual tested modules when combined with other modules are not tested for their interfaces. Therefore they may contain bugs in integrated environment. Thus, the intention here is to uncover the bugs that are present when the unit tested modules are integrated.

__/__/___

Function Testing: The main objective is to measure the quality of the functional components of the system. Verifies that the system behaves properly from the user's perspective and functions according to the requirements. It must determine if each component:
→ performs in accordance to the specification
→ responds correctly to all the conditions that may be presented by incoming events/ data.
→ moves data correctly from one business event to another.
→ is initiated in the order required to meet the business objectives of the system.

System Testing: series of different tests whose main purpose is to fully exercise a computer based system. Does not aim to test the specified function but its intention is to test the whole system on various grounds where bugs may occur.
Eg: If a software fails, how does it recover?

Acceptance Testing: Process of comparing the final system with the needs of the customer as agreed on the acceptance criteria.