**Testing**

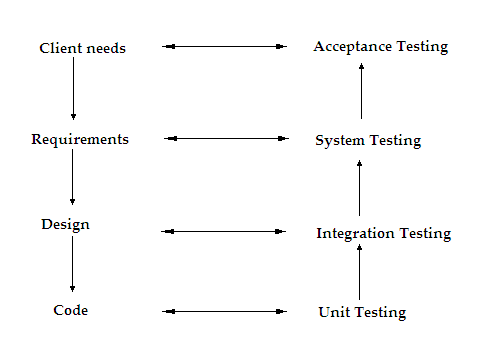
Testing is one of the most important phases in the software development activity. In software development life cycle (SDLC), the main aim of testing process is the quality; the developed software is tested against attaining the required functionality and performance.

During the testing process the software is worked with some particular test cases and the output of the test cases are analyzed whether the software is working according to the expectations or not.

The success of the testing process in determining the errors is mostly depends upon the test case criteria, for testing any software we need to have a description of the expected behavior of the system and method of determining whether the observed behavior confirmed to the expected behavior.

**Levels of Testing**

Since the errors in the software can be injured at any stage. So, we have to carry out the testing process at different levels during the development. The basic levels of testing are Unit, Integration, System and Acceptance Testing.



**Fig: 4.1 Levels of Testing**

The Unit Testing is carried out on coding. Here different modules are tested against the specifications produced during design for the modules. In case of integration testing different tested modules are combined into sub systems and tested in case of the system testing the full software is tested and in the next level of testing the system is tested with user requirement

**Unit Testing**

Unit testing mainly focused first in the smallest and low-level modules, proceeding one at a time. Bottom-up testing was performed on each module. As developing a driver program, that tests modules by developed or used. But for the purpose of testing, modules themselves were used as stubs, to print verification of the actions performed. After the lower level modules were tested, the modules that in the next higher level those make use of the lower modules were tested. Each module was tested against required functionally and test cases were developed to test the boundary values.

**Integration Testing**

Is a systematic technique for constructing the program structure, while at the same time conducting tests to uncover errors associated with interfacing as the system consists of the number of modules the interface to be tested was between the edges of the two modules. The software tested under this was incremental bottom-up approach.

Bottom-up approach integration strategy was implemented with the following steps.

* Low-level modules were combined into clusters that perform specific software sub functions.
* The clusters were then tested.

**System Testing**

Is a series of different tests whose primary purpose is to fully exercise the computer-based system? It also tests to find discrepancies between the system and its original objective, current specifications.

**Alpha Testing**

For this project alpha testing is carried out by the customer with in the organization along with the developer. The Alpha tests are conducted in a controlled manner.

**Beta Testing**

Beta testing has performed by selecting group customers. The developer is not present at the site and the user will inform the problems that are encountered. When in future problems are reported then they are rectified by software developer.

**Acceptance Testing**  Acceptance Testing has performed by the customer to whether or not of accepts the delivery of the system.

**Functional Testing**

In Functional Testing test cases are decided solely on the basis of requirements of the program or module and the internals of the program or modules are not considered for selection of test cases. This is also called Black Box Testing.

**Structural Testing**

In Structural Testing test cases are generated on actual code of the program or module to be tested. This is called White Box Testing.

**Testing Process**

A number of activities must be performed for testing software. Testing starts with test plan. Test plan identifies all testing related activities that need to be performed along with the schedule and guide lines for testing. The plan also specifies the levels of testing that need to be done, by identifying the different testing units. For each unit specified in the plan first the test cases and reports are produced. These reports are analyzed.

**Test Plan**

Test plan is a general document for entire project, which defines the scope, approach to be taken and the personal responsible for different activities of testing. The inputs for forming test plane are

* Project plan
* Requirements document
* System design

**Test Case Specification**

Although there is one test plan for entire project test cases have to be specified separately for each test case. Test case specification gives for each item to be tested. All test cases and outputs expected for those test cases.

**Test Case Execution and Analysis**

The steps to be performed for executing the test cases are specified in separate document called test procedure specification. This document specifies any specify requirements that exist for setting the test environment and describes the methods and formats for reporting the results of testing.

#### Test Approach

Testing can be done in two ways:

* Bottom up approach
* Top down approach

**Bottom up Approach**

Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded with in the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

**Top down approach**

This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module.

**Validation**

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed

**White box and black box testing:**

To meet Wikipedia's quality standards, this section may require cleanup. Please discuss this issue on the talk page, and/or replace this tag with a more specific message.

White box and black box testing are terms used to describe the point of view a test engineer takes when designing test cases. Black box being an external view of the test object and white box being an internal view. Software testing is partly intuitive, but largely systematic. Good testing involves much more than just running the program a few times to see whether it works.

Thorough analysis of the program under test, backed by a broad knowledge of testing techniques and tools are prerequisites to systematic testing. Software Testing is the process of executing software in a controlled manner; in order to answer the question “Does this software behave as specified?” Software testing is used in association with Verification and Validation.

Verification is the checking of or testing of items, including software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques as reviews, inspections, walk-through. Validation is the process of checking what has been specified is what the user actually wanted.

In order to achieve consistency in the Testing style, it is imperative to have and follow a set of testing principles. This enhances the efficiency of testing within SQA team members and thus contributes to increased productivity. The purpose of this document is to provide overview of the testing, plus the techniques.

**4.1 Test Cases**

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

+ve Test Cases:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No** | Input | Expected Behavior | **Observed**  **Behavior** | **Status** |
| 1 | Enter the details of Public Data | Name:Ram  D.O.B:06/08/1992  Address:Bobbili | After Enter details submit Generate QR-code | True |
| 2 | Enter the Private message | Enter any message | It Navigates to ECC | True |
| 3 | Show Code-Words | Observe those codeword’s and it navigate to Texture Pattern | Observe those codeword’s and it navigate to Texture Pattern | True |
| 4 | 2LQR-code Generator | P&S Navigates to 2LQR-code | P&S Navigates to 2LQR-code | True |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TesNo** | Input | Expected Behavior | **Observed**  **Behavior** | **Status** |
| 1 | Enter the details of Public Data | Name:Ram  D.O.B:06/08/1992  Address:Bobbili | After Enter details submit Not Generate QR-code | False |
| 2 | Enter the Private message | Enter any message | Not Navigates to ECC | False |
| 3 | Show Code-Words | Observe those codeword’s and it navigate to Texture Pattern | Observe those codeword’s and it navigate to Texture Pattern Not Shown | False |
| 4 | 2LQR-code Generator | P&S Navigates to 2LQR-code | P&S Navigates to Not generating 2LQR-code | False |

-ve Test Cases: