**smoke test:**A subset of all defined/planned test cases that cover the main functionality of a component or system, to ascertaining that the most crucial functions of a program work, but not bothering with finer details. A daily build and smoke test is among industry best practices. See also intake test.

**Unit Testing:** Unit Testing is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers

**integration:**The process of combining components or systems into larger assemblies.

**integration testing:**Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. See also component integration testing, system integration testing.

**functional requirement:**A requirement that specifies a function that a component or system must perform. [IEEE 610]

**functional testing:**Testing based on an analysis of the specification of the functionality of a component or system. See also black box testing.

**regression testing:**Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.

**re-testing:**Testing that runs test cases that failed the last time they were run, in order to verify the success of corrective actions.

**static testing:**Testing of a component or system at specification or implementation level without execution of that software, e.g. reviews or static code analysis.

**dynamic testing:**Testing that involves the execution of the software of a component or system.

**test strategy:**A high-level document defining the test levels to be performed and the testing within those levels for a programme (one or more projects).

**test plan:**A document describing the scope, approach, resources and schedule of intended test activities. It identifies amongst others test items, the features to be tested, the testing tasks, who will do each task, degree of tester independence, the test environment, the test design techniques and test measurement techniques to be used, and the rationale for their choice, and any risks requiring contingency planning. It is a record of the test planning process [After IEEE 829]

**test script:**Commonly used to refer to a test procedure specification, especially an automated one.

**test tool:**A software product that supports one or more test activities, such as planning and control, specification, building initial files and data, test execution and test analysis. [TMap] See also CAST.

**V-model:**A framework to describe the software development life cycle activities from requirements specification to maintenance. The V-model illustrates how testing activities can be integrated into each phase of the software development life cycle.

**validation:**Confirmation by examination and through provision of objective evidence that the requirements for a specific intended use or application have been fulfilled. [ISO 9000]

**verification:**Confirmation by examination and through the provision of objective evidence that specified requirements have been fulfilled. [ISO 9000]

**white box test design technique:**Documented procedure to derive and select test cases based on an analysis of the internal structure of a component or system.

**white box testing:**Testing based on an analysis of the internal structure of the component or system.

**data driven testing:**A scripting technique that stores test input and expected results in a table or spreadsheet, so that a single control script can execute all of the tests in the table. Data driven testing is often used to support the application of test execution tools such as capture/playback tools. [Fewster and Graham] See also keyword driven testing.

**black box testing:**Testing, either functional or non-functional, without reference to the internal structure of the component or system.

**debugging:**The process of finding, analyzing and removing the causes of failures in software.

**actual result:**The behavior produced/observed when a component or system is tested.

**expected result:**The behavior predicted by the specification, or another source, of the component or system under specified conditions.

**exception handling:**Behavior of a component or system in response to erroneous input, from either a human user or from another component or system, or to an internal failure.

**defect:**A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.

**acceptance criteria:**The exit criteria that a component or system must satisfy in order to be accepted by a user, customer, or other authorized entity. [IEEE 610]

**acceptance testing:**Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system. [After IEEE 610]

**code coverage:**An analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g. statement coverage, decision coverage or condition coverage.

**maintenance:**Modification of a software product after delivery to correct defects, to improve performance or other attributes, or to adapt the product to a modified environment. [IEEE 1219]

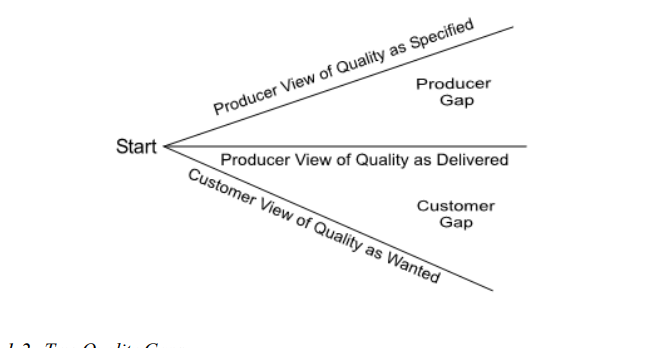
**maintainability:**The ease with which a software product can be modified to correct defects, modified to meet new requirements, modified to make future maintenance easier, or adapted to a changed environment. [ISO 9126]

**Pass:**A test is deemed to pass if its actual result matches its expected result.

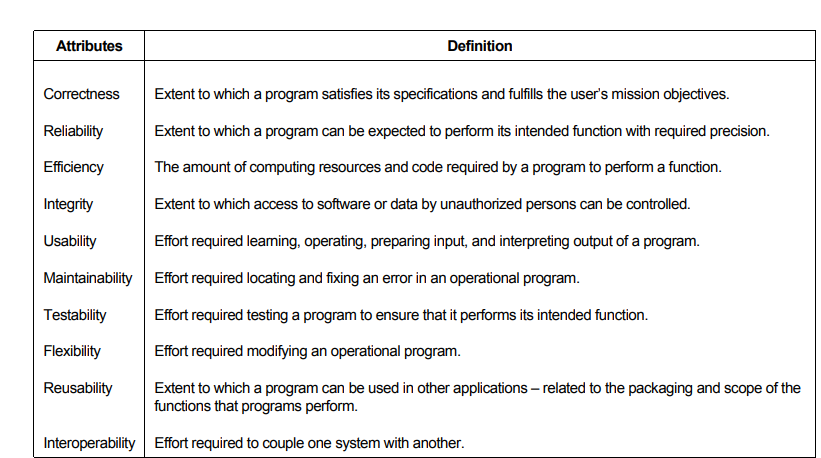
**pass/fail criteria:**Decision rules used to determine whether a test item (function) or feature has passed or failed a test. [IEEE 829]

**quality:**The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations. [After IEEE 610]

**Flow diagram Development to testing**

Example

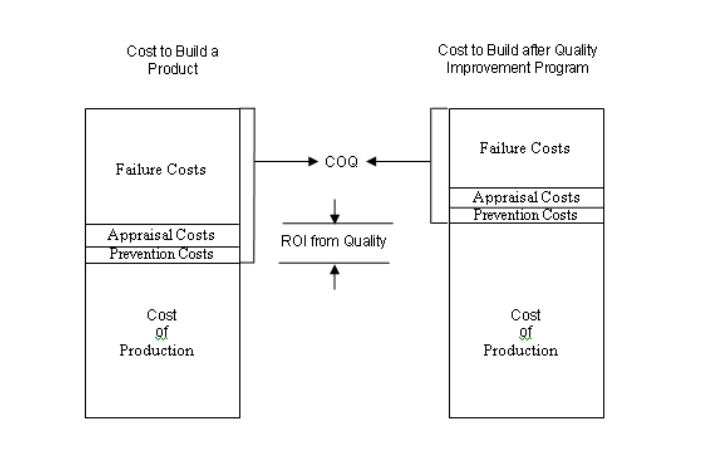
Different attributes for Software product :

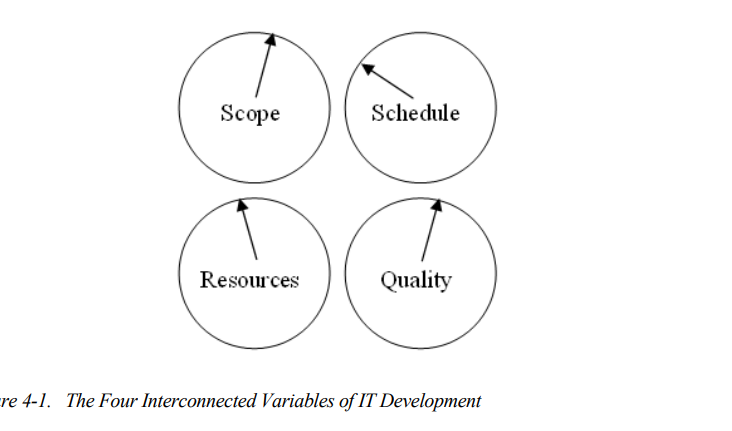


Early detection of defects , would help in reducing the effort and money.

Hence reducing the cost of product also.

Testing should be there at all levels and from the very beginning





These are four dials of software

Scope: The testing requirements which needs to be covered in test cases

Schedule: Time before which development,testing activities should be completed

Resources: Money, people and tools required to build the software

Quality :

1. Product should meet with the client requirements( BRS)
2. Product should be “fit for use”.