**Software Testing**

Software systems are an integral part of life, from business applications (e.g., banking) to consumer products (e.g., cars).Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is[Defect](https://www.guru99.com/defect-management-process.html)free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

**Why is Software Testing Important?**

**Software Testing is Important** because if there are any bugs or errors in the software, it can be identified early and can be solved before delivery of the software product. Properly tested software products ensure reliability, security and high performance which further results in time saving, cost effectiveness and customer satisfaction.

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss, and history is full of such examples.

**Typical Objectives of Testing**

For any given project, the objectives of testing may include:

• To prevent defects by evaluate work products such as requirements, user stories, design, and code

• To verify whether all specified requirements have been fulfilled

• To check whether the test object is complete and validate if it works as the users and other stakeholders expect

• To build confidence in the level of quality of the test object

• To find defects and failures by reducing the level of risk of inadequate software quality

• To provide sufficient information to stakeholders to allow them to make informed decisions, especially regarding the level of quality of the test object

• To comply with contractual, legal, or regulatory requirements or standards, and/or to verify the test object’s compliance with such requirements or standards.

The objectives of testing can vary, depending upon the context of the component or system being tested, the test level, and the software development lifecycle model. These differences may include, for example:

• During component testing, one objective may be to find as many failures as possible so that the underlying defects are identified and fixed early. Another objective may be to increase code coverage of the component tests.

• During acceptance testing, one objective may be to confirm that the system works as expected and satisfies requirements. Another objective of this testing may be to give information to stakeholders about the risk of releasing the system at a given time.

**Quality Assurance - Quality Control- Testing**

### **What is Assurance?**

Assurance is provided by organization management, it means giving a positive declaration on a product which obtains confidence for the outcome. It gives a security that the product will work without any glitches as per the expectations or requests.

### **What is Quality Assurance?**

Quality Assurance is known as QA and focuses on preventing defects. Quality Assurance ensures that the approaches, techniques, methods and processes designed for the projects are implemented correctly.

Quality assurance activities monitor and verify that the processes used to manage and create the deliverables have been followed and are operative.

Quality Assurance is a proactive process and is Prevention in nature. It recognizes flaws in the process. Quality Assurance has to be completed before Quality Control.

**Quality Assurance** is a broad term, explained as “*the continuous and consistent improvement and maintenance of process that enables the QC job*”. As follows from the definition, QA focuses more on organizational aspects of quality management, monitoring the consistency of the production process.

**What is control**

Control is to test or verify actual results by comparing it with the defined standards.

**What is Quality Control**

Quality Control is known as QC and focuses on identifying a defect. QC ensures that the approaches, techniques, methods and processes designed in the project are following correctly. QC activities monitor and verify that the project deliverables meet the defined quality standards.

Quality Control is a reactive process and is detection in nature. It recognizes the defects. Quality Control has to complete after Quality Assurance.

Through **Quality Control** the team verifies the product’s compliance with the functional requirements. It is a “*process through which a business seeks to ensure that product quality is maintained or improved and manufacturing errors are reduced or eliminated*”. This activity is applied to the finished product and performed before the product release. In terms of manufacturing industry, it is similar to pulling a random item from an assembly line to see if it complies with the technical specs.

**Testing** is the basic activity aimed at detecting and solving technical issues in the software source code and assessing the overall product usability, performance, security, and compatibility. It has a very narrow focus and is performed by the test engineers in parallel with the development process or at the dedicated testing stage (depending on the methodological approach to the software development cycle).

****

|  |  |
| --- | --- |
| **Quality Assurance** | **Quality Control** |
| It is a process which deliberates on providing assurance that quality requests will be achieved. | QC is a process which deliberates on fulfilling the quality request. |
| A QA aim is to prevent the defect. | A QC aim is to identify and improve the defects. |
| QA is the technique of managing quality. | QC is a method to verify quality. |
| QA does not involve executing the program. | QC always involves executing the program. |
| All team members are responsible for QA. | Testing team is responsible for QC. |
| QA Example: Verification | QC Example: Validation. |
| QA means Planning for doing a process. | QC Means Action for executing the planned process. |
| Statistical Technique used on QA is known as Statistical Process Control (SPC.) | Statistical Technique used on QC is known as Statistical Quality Control (SPC.) |
| QA makes sure you are doing the right things. | QC makes sure the results of what you've done are what you expected. |
| QA Defines standards and methodologies to follow in order to meet the customer requirements. | QC ensures that the standards are followed while working on the product. |
| QA is the process to create the deliverables. | QC is the process to verify deliverables. |
| QA is responsible for a full software development life cycle. | QC is responsible for [software testing life cycle.](https://www.softwaretestinghelp.com/what-is-software-testing-life-cycle-stlc/) |

**Errors, Bug, and Failures**

**Defect can be categorized into the following:**

**ERROR**: An **Error** is a mistake made in the code; that's why we cannot execute or compile code. A mistake, misconception, or misunderstanding on the part of a software developer. For example, a developer may misunderstand a design notation, or a programmer might type a variable name incorrectly – leads to an Error. It is the one which is generated because of a wrong login, loop or due to syntax. Error normally arises in software; it leads to change the functionality of the program.

**BUG**: A bug is the result of a coding error. An Error found in the development environment before the product is shipped to the customer. A programming error that causes a program to work poorly, produce incorrect results or crash. An error in software or hardware that causes a program to malfunction. Bug is the terminology of Tester.

**FAILURE**: A failure is the inability of a software system or component to perform its required functions within specified *performance requirements*. When a defect reaches the end customer it is called a Failure.  **Failure occurs when the software fails to perform in the real environment.** During development Failures are usually observed by testers or final users.

**Here are the 7 Principles of Software Testing:**

**1) Exhaustive testing is not possible**

It is not possible to test all the functionalities with all valid and invalid combinations of input data during actual testing. Instead of this approach, testing of a few combinations is considered based on priority using different techniques.

Exhaustive testing will take unlimited efforts and most of those efforts are ineffective. Also, the project timelines will not allow testing of so many combinations. Hence it is recommended to test input data using different methods like Equivalence Partitioning and Boundary Value Analysis.

**For Example**, If suppose we have an input field which accepts alphabets, special characters, and numbers from 0 to 1000 only. Imagine how many combinations would appear for testing, it is not possible to test all combinations for each input type.

The testing efforts required to test will be huge and it will also impact the project timeline and cost. Hence it is always said that exhaustive testing is practically not possible.

**2) Defect Clustering**

Defect Clustering which states that a small number of modules contain most of the defects detected. This is the application of the Pareto Principle to software testing: approximately 80% of the problems are found in 20% of the modules.

By experience, you can identify such risky modules. But this approach has its own problems

If the same tests are repeated over and over again, eventually the same test cases will no longer find new bugs.

**3) Pesticide Paradox**

Repetitive use of the same pesticide mix to eradicate insects during farming will over time lead to the insects developing resistance to the pesticide Thereby ineffective of pesticides on insects. The same applies to software testing. If the same set of repetitive tests are conducted, the method will be useless for discovering new defects.

To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.

Testers cannot simply depend on existing test techniques. He must look out continually to improve the existing methods to make testing more effective. But even after all this sweat & hard work in testing, you can never claim your product is bug-free. To drive home this point, let's see this video of the public launch of Windows 98

You think a company like MICROSOFT would not have tested their OS thoroughly & would risk their reputation just to see their OS crashing during its public launch!

**4) Testing shows a presence of defects**

Hence, the testing principle states that - Testing talks about the presence of defects and don’t talk about the absence of defects. i.e. Software Testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.

But what if you work extra hard, taking all precautions & making your software product 99% bug-free. And the software does not meet the needs & requirements of the clients.

This leads us to our next principle, which states that- Absence of Error

**5) Absence of Error - fallacy**

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement. Software testing is not merely finding defects, but also to check that software addresses the business needs. The absence of Error is a Fallacy i.e. Finding and fixing defects does not help if the system build is unusable and does not fulfill the user's needs & requirements.

To solve this problem, the next principle of testing states that Early Testing

**6) Early Testing**

Early Testing - Testing should start as early as possible in the Software Development Life Cycle. So that any defects in the requirements or design phase are captured in early stages. It is much cheaper to fix a Defect in the early stages of testing. But how early should start testing? It is recommended that you start finding the bug the moment the requirements are defined. More on this principle in a later training tutorial.

**7) Testing is context dependent**

Testing is context dependent which basically means that the way you test an e-commerce site will be different from the way you test a commercial off the shelf application. All the developed software’s are not identical. You might use a different approach, methodologies, techniques, and types of testing depending upon the application type. For instance testing, any POS system at a retail store will be different from testing an ATM machine.

**Test Activities and Tasks**

A test process consists of the following main groups of activities:

* Test planning - objectives of testing
* Test monitoring and control - comparison of actual progress against planned progress using any test monitoring metrics
* Test analysis - “what to test” in terms of measurable coverage criteria.
* Test design - “how to test?”
* Test implementation - “do we now have everything in place to run the tests?”
* Test execution
* Test completion - collect data from completed test activities to consolidate experience, testware, and any other relevant information.

Optional:

**Test work products**

Test work products are created as part of the Test Process. For each Test Process Activities Group, there are corresponding Work Products. Work products are generally documents prepared as a part of Test Activities. There is significant variation in the kinds of work products created during the Test process, how those products are organized and managed, and the terms used for those work products.

**Traceability between the test basis and test work products**

In order to implement effective test monitoring and control, it is essential to establish and maintain Traceability throughout the test process between each element of the test basis and the various test work products associated with that element.

Test Coverage evaluation is done using Traceability. Also, using Traceability, we can:

* Examine the impact of changes
* Make Testing Auditable
* Fulfill IT governance criteria
* Improve the understanding of Test Status and Summary Reports. For example, information related to requirement mapped test cases status like a number of the pass, fail, or pending test cases.
* To evaluate if we have missed any aspects to be tested or documented.
* Communicate the technical aspect of Testing to stakeholders
* Provide information about product quality
* Share information regarding Process capability
* Convey Project Progress against Business Goals

Organizations use Test Management tools or project management tools or build their custom solutions to organize the work products and provide the information traceability they require.