**Quality Assurance**

**What is Quality?**

Quality is defined as the product or services that should be "fit for use and purpose."

Quality is all about meeting the needs and expectations of customers concerning functionality, design, reliability, durability, and price of the product.

**What is Assurance?**

Assurance is a positive declaration on a product or service. It is all about the product which should work well. It provides a guarantee which would work without any problem according to expectations and requirements.

**What is Quality Assurance?**

Quality Assurance is also known as **QA Testing**. ***QA*** is defined as an activity to ensure that an organization is providing the best product or service to the customers.

Software Quality Assurance seems it is all about evaluation of software based on functionality, performance, and adaptability; however software quality assurance goes beyond the quality of the software, it also includes the quality of the process used to develop, test and release the software.

Software Quality assurance is all about the Software Development lifecycle that includes requirements management, software design, coding, testing, and release management.

Quality Assurance is the set of activities that defines the procedures and standards to develop the product.

Quality Assurance is a systematic way of creating an environment to ensure that the software product being developed meets the quality requirements. This process is controlled and determined at the managerial level. It is a preventive process whose aim is to establish the correct methodology and standard to provide a quality environment to the product being developed. Quality Assurance focuses on process standard, projects audit, and procedures for development. QA is also known as a set of activities designed to evaluate the process by which products are manufactured.

QA focused on improving the processes to deliver Quality Products.

**What is the Quality Attribute of software?**

The following six characteristics can define the quality of the software:

**1. Functionality**

Quality of software is defined as how effectively the software interacts with other components of the system. The software must provide appropriate functions as per requirement, and these functions must be implemented correctly.

**2. Reliability**

It is defined as the capability of the software to perform under specific conditions for a specified duration.

**3. Usability**

Usability of software is defined as its ease of use. Quality of the software is also identified as how easily a user can understand the functions of the software and how much efforts are required to follow the features.

**4. Efficiency**

The efficiency of the software is dependent on the architecture and coding practice followed during development.

**5. Maintainability**

Maintainability is also one of the significant factors to define the quality of the software. It refers to identify the fault and fix in the software. It should be stable when the changes are made.

**6. Portability**

Portability of the software, defined as how easily a system adapts to changes in the specifications. Quality of the software is also determined by the portability of the system how easy it is to install the software and how easy it is to replace a component of the order in a given environment.

To ensure about a software score well on these quality attribute, we need the following software Quality Assurance.

**Levels of Quality Assurance**

**There are three levels of quality assurance: testing, validation, and certification.**

1. In system testing, the common view is to eliminate program errors. The is extremely difficult and time-consuming, since designers cannot prove 100 percent accuracy. Therefore, all that can be cone is to put the through a "fail test" cycle-determine what will make it fail. A successful test, then, is one that finds errors. The test strategies discussed earlier are used in system testing.
2. System validation checks the quality of the software in both simulated and live environments. First the software goes through a phase (often referred to as alpha testing) in which errors and failures based on simulated user re4uirements are verified and studied. The modified software is then subjected to phase two (called beta testing) in the actual user's site or a live environment. The system is used regularly with live transactions. After a scheduled time, failures and error are documented and final' correction, and enhancements are made before the package is released for use.
3. The third level of quality assurance is to certify that the program or software package is current and conforms to standards. With a growing trend toward purchasing ready-to-use software, certification has become more important. A package that is certified goes through a team of specialists who test, review, and determine how well it meets the vendor's claims. Certification is actually issued after the package passes the test. Certification, however, does not assure the use' that it is the best package to adopt; it only attests that it will perform what the vendor claims.

**Audit Trail**

Audit Trails An important function of system controls is providing for an audit trail. An audit trail is a routine designed to allow the analyst, user or auditor to verify a process or an area in the new system.

**Definition of Audit trail** A feature of data processing systems that allows for the study of data as processed from step to step, an auditor may then trace all transactions that affect an account. In a manual system, the audit trail includes journals, ledgers and other documents used by auditor to trace transactions. In a computerized system, record content and format frequently make it difficult to trace a transaction completely. Some reasons are the following:

Files stored on the tape or disk can be read only by a computer, which limits the auditing function. A data dump is possible, though, to compare the data against a data map. 2. Direct data entry eliminates the physical documentation for an audit program. 3. Data processing activities are difficult to observe, since they take place within the computer system. For the audit trail to show its impact a detailed file of the transactions need to be maintained. During evaluation of a system following steps should be considered.

1. Define the control objectives as separate design and test requirements. Input preparation and transmission by the user are important control areas that are viewed with an emphasis on audit trails and adequate documentation during testing.

2. Examine budget costs to see whether system testing is within the limits.

3. Review specifications. The auditor should evaluate program acceptance test specifications and assist the programmer in developing test standards, levels of testing and actual test conditions.

It is the auditor’s responsibility to build controls into candidate systems to ensure reliability, integrity and confidence of the users at all levels. The auditor should be called in during design as well as testing so that suggestion can be considered before implementation. Including the auditor in the system development team makes it easy for monitoring testing procedures and considers the acceptance of new controls to replace those changed by the new design.

In summary, the primary purpose of auditing is to check that controls built into the design of candidate systems ensure its integrity. Audit considerations must be incorporated at an early

**Definition of 'Software Maintenance'**

**Definition:**

Software maintenance is a part of Software Development Life Cycle. Its main purpose is to modify and update software application after delivery to correct faults and to improve performance. Software is a model of the real world. When the real world changes, the software requires alteration wherever possible.

**Description:**

Software maintenance is a vast activity which includes optimization, error correction, and deletion of discarded features and enhancement of existing features. Since these changes are necessary, a mechanism must be created for estimation, controlling and making modifications. The essential part of software maintenance requires preparation of an accurate plan during the development cycle. Typically, maintenance takes up about 40-80% of the project cost, usually closer to the higher pole. Hence, a focus on maintenance definitely helps keep costs down.

Software Maintenance Processes are:   
  
• The SM process includes a maintenance plan which contains software preparation, problem identification and find out about product configuration management.   
  
• The problem analysis process includes checking validity, examining it and coming up with a solution and finally getting all the required support to apply for modification.   
  
• The process acceptance by confirming the changes with the individual who raised the request.   
  
• The platform migration process, which is used if software is needed to be ported to another platform without any change in functionality.   
  
Some software points that affect maintenance cost include:   
  
• Structure of Software Program   
  
• Programming Language   
  
• Dependence on external environment   
  
• Staff reliability and availability

**Types of maintenance**

In a software lifetime, type of maintenance may vary based on its nature. It may be just a routine maintenance tasks as some bug discovered by some user or it may be a large event in itself based on maintenance size or nature. Following are some types of maintenance based on their characteristics:

1. **Corrective Maintenance** - This includes modifications and updations done in order to correct or fix problems, which are either discovered by user or concluded by user error reports.
2. **Adaptive Maintenance** - This includes modifications and updations applied to keep the software product up-to date and tuned to the ever changing world of technology and business environment.
3. **Perfective Maintenance** - This includes modifications and updates done in order to keep the software usable over long period of time. It includes new features, new user requirements for refining the software and improve its reliability and performance.
4. **Preventive Maintenance** - This includes modifications and updations to prevent future problems of the software. It aims to attend problems, which are not significant at this moment but may cause serious issues in future.

**Cost of Maintenance**

Reports suggest that the cost of maintenance is high. A study on estimating software maintenance found that the cost of maintenance is as high as 67% of the cost of entire software process cycle.



On an average, the cost of software maintenance is more than 50% of all SDLC phases. There are various factors, which trigger maintenance cost go high, such as:

**Real-world factors affecting Maintenance Cost**

* The standard age of any software is considered up to 10 to 15 years.
* Older softwares, which were meant to work on slow machines with less memory and storage capacity cannot keep themselves challenging against newly coming enhanced softwares on modern hardware.
* As technology advances, it becomes costly to maintain old software.
* Most maintenance engineers are newbie and use trial and error method to rectify problem.
* Often, changes made can easily hurt the original structure of the software, making it hard for any subsequent changes.
* Changes are often left undocumented which may cause more conflicts in future.

**Hardware Selection Criteria**

* Hardware must support current software as well as software planned for procurement over the next planning interval [*year, 18 months, three years*]
* Hardware must be compatible with existing or planned networks
* Hardware must be upgradeable and expandable to meet the needs of the next planning interval
* Hardware warranties must be of an appropriate length
* Hardware maintenance must be performed by [*local/remote vendor, in-house personnel*]
* Whenever feasible, hardware standards will dictate procurement of like brands and configurations to simplify installation and support
* Routine assessments of installed infrastructure will feed an upgrade/replace decision process

**Software Selection Criteria**

* Software must be compatible with current and future hardware over the next planning interval
* Software maintenance and warranties must be of appropriate length and cost
* Sotware help desk must be maintained by [*vendor, third party, in-house personnel*]
* Software must be standardized throughout the business to improve purchasing power, simplify training, and facilitate support
* Software must comply with current standards set by technology leadership
* Software must support and enhance business goals

<https://www.1000sourcecodes.com/2012/06/sad-procedure-for-hardware-software.html>