**Software Quality Assurance Plan**

Abbreviated as SQAP, the software quality assurance plan comprises of the procedures, techniques, and tools that are employed to make sure that a product or service aligns with the requirements defined in the SRS (software requirement specification).



The plan identifies the SQA responsibilities of a team, lists the areas that need to be reviewed and audited. It also identifies the SQA work products.

* **Goals and Objectives of SQA:**
* **Small to Zero Defects after Installation** – One of the biggest goals of SQA is to prevent any possible defects when the output is made. Developers and engineers have to use universally approved steps to ensure that the program was built up to expectations but also to prevent errors in the system. Although some standards allow as much as .04 errors in the system, zero-error is still the system’s target. When there’s zero-error, the program is more likely to have zero crash scenarios. The ability to handle stress of a program is different from the errors it has but crashes usually comes from defects so prevention of defects will most likely yield a continuously working application.
* **Customer Satisfaction** – Everything else will be just nothing if the customers don’t like what they see. Part of SQA is to ensure that software development was made according to their needs, wants and exceeding their expectations. Even if the bugs and errors are minimized by the system, customer satisfaction is more important and should be emphasized.
* **Well Structured** – SQA takes care of the stages of application construction. Anyone could be easily build an application and launch it in their environment without any glitches. However, not everyone could easily build an application that could be understood well. SQA ensures that each application are built in an understandable manner. Their applications could easily be transferred from one developer to another.

**The SQA plan document consists of the below sections:**

1. Purpose section
2. Reference section
3. Software configuration management section
4. Problem reporting and corrective action section
5. Tools, technologies and methodologies section
6. Code control section
7. Records: Collection, maintenance and retention section
8. Testing methodology

SQA Activities

***Given below is the list of SQA activities:***

**#1) Creating an SQA Management Plan:**

The foremost activity includes laying down a proper plan regarding how the SQA will be carried out in your project.

Along with what SQA approach you are going to follow, what engineering activities will be carried out, and it also includes ensuring that you have a right talent mix in your team.

**#2) Setting the Checkpoints:**

The SQA team sets up different checkpoints according to which it evaluates the quality of the project activities at each checkpoint/project stage. This ensures regular quality inspection and working as per the schedule.

**#3) Apply software Engineering Techniques:**

Applying some software engineering techniques aids a software designer in achieving high-quality specification. For gathering information, a designer may use techniques such as interviews and FAST (Functional Analysis System Technique).

Later, based on the information gathered, the software designer can prepare the project estimation using techniques like WBS (work breakdown structure), SLOC (source line of codes), and FP(functional point) estimation.

**#4) Executing Formal Technical Reviews:**

An FTR is done to evaluate the quality and design of the prototype.

In this process, a meeting is conducted with the technical staff to discuss regarding the actual quality requirements of the software and the design quality of the prototype. This activity helps in detecting errors in the early phase of SDLC and reduces rework effort in the later phases.

**#5) Having a Multi- Testing Strategy:**

By multi-testing strategy, we mean that one should not rely on any single testing approach, instead, multiple types of testing should be performed so that the software product can be tested well from all angles to ensure better quality.

**#6) Enforcing Process Adherence:**

This activity insists the need for process adherence during the software development process. The development process should also stick to the defined procedures.

**This activity is a blend of two sub-activities which are explained below in detail:**

**(i) Product Evaluation:**

This activity confirms that the software product is meeting the requirements that were discovered in the project management plan. It ensures that the set standards for the project are followed correctly.

**(ii) Process Monitoring:**

This activity verifies if the correct steps were taken during software development. This is done by matching the actually taken steps against the documented steps.

**#7) Controlling Change:**

In this activity, we use a mix of manual procedures and automated tools to have a mechanism for change control.

By validating the change requests, evaluating the nature of change and controlling the change effect, it is ensured that the software quality is maintained during the development and maintenance phases.

**#8) Measure Change Impact:**

If any defect is reported by the QA team, then the concerned team fixes the defect.

After this, the QA team should determine the impact of the change which is brought by this defect fix. They need to test not only if the change has fixed the defect, but also if the change is compatible with the whole project.

For this purpose, we use software quality metrics which allows managers and developers to observe the activities and proposed changes from the beginning till the end of SDLC and initiate corrective action wherever required.

**#9) Performing SQA Audits:**

The SQA audit inspects the entire actual SDLC process followed by comparing it against the established process.

It also checks whatever reported by the team in the status reports were actually performed or not. This activity also exposes any non-compliance issues.

**#10) Maintaining Records and Reports:**

It is crucial to keep the necessary documentation related to SQA and share the required SQA information with the stakeholders. The test results, audit results, review reports, change requests documentation, etc. should be kept for future reference.

**#11) Manage Good Relations:**

In fact, it is very important to maintain harmony between the QA and the development team.

We often hear that testers and developers often feel superior to each other. This should be avoided as it can affect the overall project quality.

**Elements of Software Quality Assurance**

There are 10 essential elements of SQA which are enlisted below for your reference:

1. Software engineering Standards
2. Technical reviews and audits
3. Software Testing for quality control
4. Error collection and analysis
5. Change management
6. Educational programs
7. Vendor management
8. Security management
9. Safety
10. Risk management

SQA Techniques

There are several techniques for SQA. Auditing is the chief technique that is widely adopted. However, we have a few other significant techniques as well.

Various SQA Techniques include:

* Auditing: Auditing involves inspection of the work products and its related information to determine if the set of standard processes were followed or not.
* Reviewing: A meeting in which the software product is examined by both the internal and external stakeholders to seek their comments and approval.
* Code Inspection: It is the most formal kind of review that does static testing to find bugs and avoid defect growth in the later stages. It is done by a trained mediator/peer and is based on rules, checklist, entry and exit criteria. The reviewer should not be the author of the code.
* Design Inspection: Design inspection is done using a checklist that inspects the below areas of software design:
  + General requirements and design
  + Functional and Interface specifications
  + Conventions
  + Requirement traceability
  + Structures and interfaces
  + Logic
  + Performance
  + Error handling and recovery
  + Testability, extensibility
  + Coupling and cohesion
* Simulation: Simulation is a tool that models the real-life situation in order to virtually examine the behavior of the system under study.
* Functional Testing: It is a QA technique which verifies what the system does without considering how it does. This type of [black box testing](https://www.softwaretestinghelp.com/black-box-testing/) mainly focuses on testing the system specifications or features.
* Standardization: Standardization plays a crucial role in quality assurance. It decreases the ambiguity and guesswork, thus ensuring quality.
* Static Analysis: It is a software analysis that is done by an automated tool without actually executing the program. This technique is highly used for quality assurance in medical, nuclear and aviation software. Software metrics and reverse engineering are some popular forms of static analysis.
* Walkthroughs: Software walkthrough or code walkthrough is a kind of peer review where the developer guides the members of the development team to go through the product and raise queries, suggest alternatives, make comments regarding possible errors, standard violations or any other issues.
* Path Testing: It is a [white box testing technique](https://www.softwaretestinghelp.com/white-box-testing-techniques-with-example/) where the complete branch coverage is ensured by executing each independent path at least once.
* Stress Testing: This type of testing is done to check how robust a system is by testing it under heavy load i.e. beyond normal conditions.
* Six Sigma: Six Sigma is a quality assurance approach that aims at nearly perfect products or services. It is widely applied in many fields including software. The main objective of six sigma is process improvement so that the produced software is 99.76 % defect free.