Software Quality Management: A Process to Improve the Quality of Software Project Using SQA

[1]Mr. Chinmaya Dash, [2] Dr. D Ramesh , [3] Mr. Prakash C Behera

[1,3]Department of Computer Science, St. Claret College, Bangalore, Karnataka, India, [2]Department of MCA, SSIT, Tumkur, Karnataka, India

***Abstract*- In today's Scenario, successful implementation of information technology projects is a critical strategic and competitive necessity for firms in all industrial sectors. Due to cost overruns, schedule delays, unfilled requirements and poor quality, IT projects are not successfully facilitating standard. There are some practices and success factors for maintaining the standard of projects but projects still continue to fail. To overcome systemic causes of project failure, a unified definition of software quality management (SQM) is proposed. We use this definition to develop and present an approach to SQM which develop and manage the quality of a software to make sure the product satisfies the user.**

In this article, it is highlighted that how to combine software quality life cycle with project management life cycle & software development life cycle to generate better quality and outcomes. This article takes a different approach by examining how software quality assurance practices can impact project outcomes. Software quality management (SQM) plays a critical role in the software development lifecycle (SDLC) and can impact a project's overall success.

***Keyword*: SQM, SQA, SQP, SQC and PMLC**

I. INTRODUCTION

Today's scenario, it is really difficult to implement successful IT projects which is a critical strategic method for entire industrial sectors. This is even more important in times of scarce resources needed for other competing strategic initiatives important to a firm. A lot of software projects still fail to deliver on time and within target costs and other necessary specifications.

Software quality management (SQM) is a management process the goal of which is to develop and manage the quality of a software to make sure the product satisfies the user. SQM is a quality management culture in an organization where quality is viewed and maintained by everyone. Software Quality Management (SQM) includes different layer such as Software Quality Assurance (SQA) layer, Software Quality Plan (SQP) layer and Software Quality Control (SQC) layer.

The **Software quality assurance (SQA)** is a monitoring process which is used to ensure the quality in entire software development lifecycle process. It is also a continuous assessment mechanism which facilitates certain procedures for project development with specific standards along with documentation. The procedures should be used to assure quality outcome (zero defects) and project success.

The **Software Quality Plan (SQP)** is an initial project level quality plan for declaring project commitment and SQP contain quality goals to be achieved, expected risks and risk management.

The **Software Quality Control (SQC)** ensures in-process that both SQA and SQP are being followed by the development teams.

II. PROJECT MANAGEMENT FAILURES

Sl No.	*Reasons for Failure*
1	Planning: Insufficient planning
2	Scope: Poorly defined and requirement and scope
3	Stakeholder: Ineffective stakeholder management, Lack of user and stakeholder involvement
4	Commitment: Lack of executive support or project sponsorship
5	Time: Poor estimation and scheduling, unrealistic deadlines
6	Cost: Unrealistic budget
7	Quality and testing: Elimination of quality assurance practice, inadequate testing or reduction of testing time within the software development lifecycle
8	Resources: Insufficient qualified resource
9	Communication: Poor and ineffective communication
10	Risk: Insufficient risk planning and risk management
11	Politics: Stakeholder politics, user politics, corporate politics
12	Project manager: Poor project management or project manager
13	Process: Undefined development process or lack of adherence to process standard, failure to implement best practices and lesson learned
14	Product: Speed to market, desire for innovation, correct errors
15	Project goal: Unrealistic project goal
16	Change control: Poorly managed change control
17	Technology: Change in technology

Table 1. Common reasons for project failure

The failure of software projects occur due to target cost or time and quality related problems. There are some challenges related to project development such as technology used for project development, the process used

for project development, scope and complexity of developed projects, and risk of developed projects. Project success requires carefully followed processes that assure quality at every phase of the project and development lifecycles. Quality assurance practices can minimize project failures by providing checks throughout this process.

III. SOFTWARE QUALITY ASSURANCE

A software quality assurance is a mechanism to assure the acceptance of quality of software along with an organization's procedures and standards. It is process which is integrated with project management as well as the project development lifecycles to check control mechanism along with software standards and procedures. The objective of SQA is to reduce risk and improve quality of the project of software with time and cost constraints.

Fig 1: Software Quality Assurance

IV. SQA PROCESS AND ROLE OF SQA TEAMS

The **SQA** is a monitoring process which is used to ensure the quality in entire software development lifecycle process. It is also a continuous assessment mechanism which facilitates certain procedures for project development with specific standards along with documentation. The procedures should be used to assure quality outcome (zero defects) and project success. At a high level, the function of SQA is to perform the following:

• **Software project planning:** Quality practices should be planned in advance which can be implemented further.

• **User requirement:** Requirements should be checked in entire project development process to satisfy user needs.

• **Design process & Coding:** Certain methodologies are followed in design process. Coding standard and guidelines must be established and implemented.

• **Software integration and Testing:** Software integration and testing should be planned and compiled as per requirement.

• **Conduct random and scheduled audits:** Perform SQA audits to assure the necessary controls are in place.

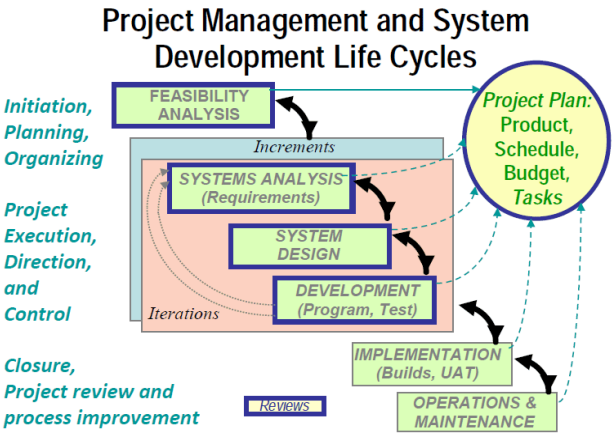

Fig 1: PMLC & SDLC

The SQA process consists of a variety of phases with specific activities. These activities should be performed by a SQA team which is responsible for software quality assurance planning, analysis, and reporting. SQA is most effective when it reports up through a separate management team so they can remain committed to the process and remain objective to the deliverable. The responsibilities of the SQA team include review of documentation for completeness and adherence to standards, participation in inspections, review of test results, and periodic audits of controls.

V. SQA PROCESS:

SQA processes provide assurance throughout the project or software management and development lifecycle. The combined project management life cycle (PMLC) and systems development life cycle (SDLC) consists of eight unique phases – initiation, planning, analysis, design, development, testing, implementation, and closing. Each SQA life cycle phase contains a feedback loop which provides information regarding issues found during SQA activities and ensures improvement.

VI. SQA LIFE CYCLE PHASES

A. Project initiation:

It is a project management function. This phase is the official start or formal kickoff of a project. The corresponding SQA phase, SQA initiation notifies the SQA team that a new project is being initiated. One of the key outcomes of this phase is to define the quality control and audit processes for the project and to ensure that the proper controls are in place.

Inputs

- Feasibility Analysis
- Business Case Document

Outputs

- Project Charter
- Preliminary Scope Statement

Controls

- Verify the appropriate outputs have been completed.

- Check the content of the outputs for correctness, consistency, and completeness (3 Cs).

During project initiation, the stakeholders, project objectives, and preliminary scope of the project or software are identified and documented. The controls in this phase include assuring that the documents must be verified for correctness, consistency, and completeness.

B. Project Planning:

Project planning is a project management function which involves deciding in advance what do, how to do it, when to do it, and who will do it. This phase includes which processes should be performed to develop the project development plan and identify the schedule which occurs within the project development. The team should finalize which organizational processes will be used.

Inputs

- Project plan
- Standards and Procedures Manual

Outputs

- Project Management Plan
- SQA Plan and Metrics
- Project plans to represent the core knowledge areas (Project Scope Management Plan, Cost Management Plan, Communication Plan, Risk Management Plan, Procurement Plan)
- Change management plan
- Work Breakdown Structure (WBS) and WBS dictionary
- Project Schedule with resource requirements
- Roles and Responsibilities

Controls

- Verify the appropriate outputs
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).

The SQA plan documents the standards, practices and metrics are used during the overall SQA process. It also includes the reviews and audit practice which will be performed. The appropriate documents must be prepared by the SQA team. The feedback should be provided to the software project team to improve the quality further. The primary function of the SQA team during this phase is to evaluate whether the nature and extent of planning are appropriate for the type of software being developed.

C. Project Analysis:

It is an SDLC phase in which the project team gathers and documents formal user requirements. The major steps in this phase are defining, reviewing and approving the requirements. Wrong requirements lead to lack of quality. So the appropriate control procedures are implemented to ensure quality. In this phase, the SQA team assures that the appropriate documents have been created based on the project or software analysis.

Inputs

- Standards and Procedures Manual

Outputs

- Requirements Definition and Baseline
- Requirements Traceability Matrix
- Software Development Plan
- High Level Design Plan

Controls

- Software Requirements Review
- Preliminary Design Review
- Verify the appropriate outputs have been completed.
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).
- Verify the requirements to ensure testability and feasibility.

Requirements verification reviews of the requirements which establishes standard of the requirement. If a requirement is not tested then it has not been defined properly. After reviewing the requirements, those should be approved by the stakeholders.

Once approved, a baseline is established. This baseline is used to track changes to the requirements in order to improve processes along the way and provide an audit trail of which requirements were changed.

D. Project Design:

In this phase, the information is translated into a blueprint from the analysis phase which can be used for developing the project or software. During the design process, it is determined which standards will be used as well as the design process that the software team will use to perform the work.

The objective of design assurance is to assure the processes being used to create the design conform to standards and that design is verified against the established requirements. Once a design process is chosen by the software design team, the SQA team reviews the selected design process in order to ensure compliance with organizational policies, internal software standards, or externally imposed standards. Quality metrics include lines of code and test case coverage.

Inputs

- Requirements Definition and Baseline
- Software Development Plan
- High Level Design Plan
- Requirements Traceability Matrix

Outputs

- Detailed Design Plan and Design Process
- Metrics Definitions to be used in the process (Product, Process and Quality Metrics)

Controls

- Design walkthrough and inspection
- Evaluation of the design process to ensure planned methodologies are followed
- Check the design process for adherence to organizational policies and standards.

- Verify the appropriate outputs have been completed.
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).

In this phase, certain control mechanisms are implemented to adhere the defined process and assure a complete design cycle. The SQA team is concerned to determine whether the programmers are used a systematic approach for designing the project or not.

E. Project Development:

During development, design is translated into code. At this stage test plans and test cases are developed so that when the software is implemented testing can be done. The SQA team works on assuring that the design process is if high quality and that testing plans are appropriate for the implementation.

Inputs

- Standards and Procedures Manual
- Software Development Plan
- Detailed Design Plan

Outputs

- Code (testable software product)
- Test Plan and Test Cases
- Metrics

Controls

- Verify unit testing
- Conduct random audits through code walkthroughs, inspections, and peer reviews
- Audit Test Plan
- Check the code to ensure adherence to standards
- Verify the development process for adherence to organizational policies and standards.
- Verify the appropriate outputs
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).

Using test cases, undiscovered errors can be detected. The testing team reviews the requirements and design documents and at the same time the team should document errors which encounter with the design. Therefore, this is one of the reasons why controls for complete requirements are necessary. It is easier to check errors before the code has been developed completely. The controls in this phase are designed to review the code to catch defects, even before testing begins and to capture metrics that were defined during the planning and design phases.

F. Project Testing:

The SQA team should evaluate the performance of unit testing. One of the biggest risks is developing a product that does not meet the stated requirements or developing a product full of defects. The development controls and assurance activities are necessary to mitigate these risks and detect defects early and the risks associated with software development are diminished. The goal of testing in SDLC is to find and document defects.

A Software Defect is a condition in a software product or project which does not meet a software requirement as stated in the requirement specifications or end-user expectations which may not be specified but are reasonable. Testing activities include integration testing, automated testing, regression testing and performance testing.

Inputs

- Test Plan and Test Cases

Outputs

- Test Summary Report
- Implementation Plan
- Metrics

Controls

- Review Test Summary Report
- Review Testing procedures
- Review Implementation Plan
- Verify the appropriate outputs
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).

The foundations of software testing include defining test processes, test cases and test plans, testing techniques, methodologies, tools, standards, and testers. After the completion of testing process, the results of that process should be documented which can be further reviewed by the developers and stakeholders. After reviewing the document, a test summary report must be generated. This important step is necessary before making a decision on whether or not to proceed with implementation. The test summary report can also highlight risk factor.

The purpose of the controls in this phase is to review the testing results to ensure complete and rigorous testing procedures were followed and to assure metrics have been captured.

G. Project Implementation:

The SQA team should review the implementation plan along with the changed management and ensure that testing of the software or project during development phase is completed and satisfactory to the users and stakeholders.

Inputs

- Implementation Plan

Outputs

- Implemented Software

Controls

- Product Release Review
- Verification the appropriate outputs
- Check the content of the outputs for the 3 Cs (correctness, consistency, and completeness).

The primary control in this phase is the product release review. The product release review ensures that while developing the software or project, new defects should not be found. Both the project development team and the SQA

team should document entire project activities and inform the stakeholders about the project completion.

H. Project Closing:

In this phase, project documentation must be reviewed through product release review. It is necessary to take the time to understand what went right and what went wrong so as not to repeat the same mistakes. During project or software development, all the metrics must be collected which can help to identify best practices for future projects.

Inputs

- Metrics
- Documentation from prior phases

Outputs

- SQA Summary Report
- Metrics

Controls

- Verify the appropriate outputs have been completed and delivered.
- Check correctness, consistency and completeness

VII. IMPLICATIONS FOR PRACTICE

SQA process is a complicated process which needs commitment and support from the entire SQA team. The term process has a negative reputation in the world of IT development. SQA process must include metrics to enhance the future improved development of software products or projects. All stakeholders need to understand and agree upon the role of the SQA process and team.

Fig 3: Perform Quality Assurance

VIII. CONCLUSION

In this topic, we have discussed the reasons why projects fail and made a case for why a software quality management process is necessary to mitigate risk of failure and reduce failure rates. We have proposed a consolidated definition of SQM and developed a process for assuring software quality that encompasses the whole software development and project management life cycles using SQA. Certain basic items must be checked such as

- Purpose of plan and its scope
- Organization structure and SQA task
- Project documents, user and technical document
- Standards, practices and conventions
- Review and audit
- Test plan and procedure
- Error reporting and correcting methods
- Tools, code and supplier control
- Record collection and maintenance
- Risk management

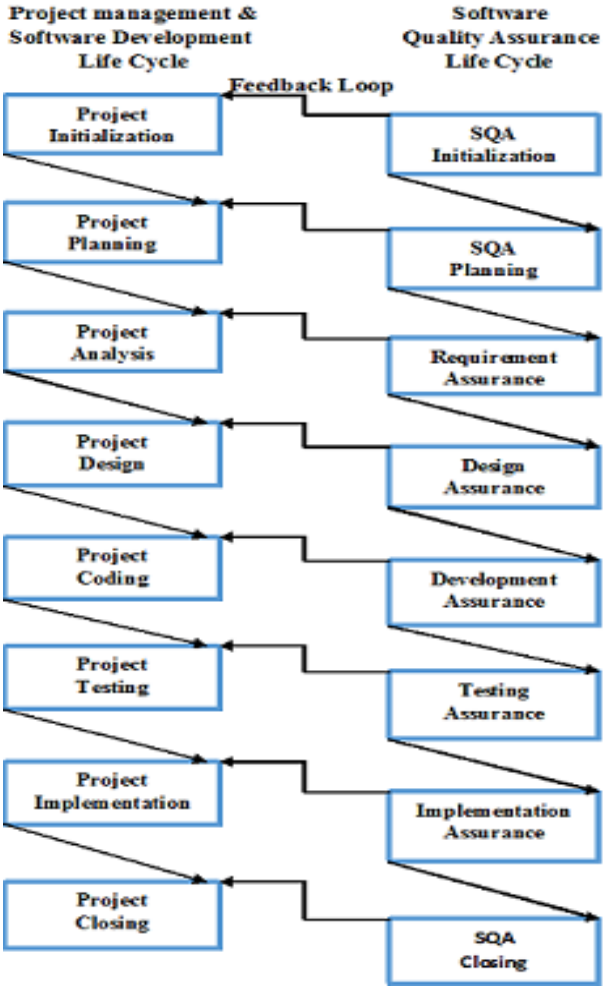

Fig 4: Combined PMLC, SDLC and SQALC

IX. REFERENCES

[1] Agarwal, R. Nayak, P., Manickam, M., Suresh, P., & Modi, N. (2007). Virtual quality assurance facilitation model. *IEEE International Conference on Global Software Engineering (ICGSE 2007)*, IEEE Computer Society.

[2] Baker, E. R. (2001). Which way, SQA? *IEEE Software*, January/February 2001, 16-18.

[3] Boehm, B., & Basili, V. R. (2001). Software defect reduction top 10 list. *Computer*, *34*(1), 135-137.

[4] Brown, S. A., Chervany, N., L., & Reinicke, B. A. (2007). What matters when introducing new information technology. *Communications of the ACM*, *50*(9), 91-96

[5] Charette, R. N. (2005). Why software fails. *IEEE Spectrum*, September, 2005, *42*.

[6] Christensen, M. J., & Thayer, R. H. (2001). *The project manager's guide to software engineering best practices*. Los Alamitos, CA: IEEE Computer Society.

[7] Galin, D. (2004). *Software quality assurance: From theory to implementation*. Harlow, UK: Pearson Education Limited.

[8] Murugesan, S. (1994, Dec. 21-22). Attitude towards testing: A key contributor to software quality. *IEEE's Proceedings of 1st International Conference on Software Testing, Reliability and Quality Assurance*, (pp. 111-115).

[9] Nelson, R. R. (2007). IT project management: Infamous failures, classic mistakes, and best practices, *MIS Quarterly Executive*, *6*(2), June 2007, 67-78.

[10] Pressman, R. S. (6th Ed.) (2005). *Software engineering: A practitioner's approach,* Boston: McGraw Hill.s