A FRAMEWORK-BASED APPROACH FOR RELIABILITY & QUALITY ASSURANCE OF SAFETY-CRITICAL SOFTWARE

Ankur Pandit
M.Tech, Final Year, Computer Science & Engineering
Lakshmi Narain College of Technology
Bhopal, India

Abstract- **With in the complex system development throughout the industries, Software has taken on a new, enhanced role and now directly impacts not only product success, but also the safety. Software Reliability & Quality Assurance (SRQA) for *Safety-Critical Software (SCS)* having the key role in mission success. The term *Safety-Critical Software* means software systems whose failure may lead to loss of life or severe injury like software used for missile, satellite, cancer radiation therapy machine etc. Every country now a day's emphasize on faster approach for developing mission. SCS involves high risk in design, development and installation. Also it is responsible for controlling, monitoring number of hardware systems inside a system. Thereby making it more important than ever to ensure the reliability and quality of software products. SRQA covers all stages of the software development process, with specific activities to assure both the processes used and the product development. In this paper a framework-based approach based on standards of reliability and quality is proposed for SRQA of SCS.**

Keywords- *Safety-Critical Software*, quality and reliability assurance, high risk, framework-based approach.

I. INTRODUCTION

Within the mission critical system development, software is encountered as crucial element in mission success. SRQA for SCS are designed and implementing differently from those of other, more tangible, physical system elements. The methods need to be developing and implement well by the program managers. In such cases the, ignorance is far from bliss; it is dangerous. As famous proverb says "Prevention is better than cure" looking better related to Reliability & Quality Assurance. SRQA linked to mission success in the long term.

Safety-Critical Software is software whose use in a system can result in unacceptable risk (for Criteria of SCS, Ref Section *C. Safety, pg. 4*). SCS includes software whose operation or failure to operate can lead to a hazardous state, software intended to recover from hazardous states, and software intended to mitigate the severity of an accident (IEEE) [1].

Many more examples in which the system cause loss of man and material due to poor Reliability & Quality Assurance of the software.

In this paper a framework-based approach based on standards of Reliability & Quality Assurance for SCS is proposed. This framework-based approach provides Software Reliability and Quality activities that should conduct throughout the project life cycle to come out with the quality product.

Note- This paper is only concern with Software Safety not with System Safety.

II. DEFINATIONS

A. Software Defined

The *IEEE* definition of software, which is almost identical to the ISO definition (ISO, 1997, Sec. 3.11 and ISO/IEC 9000-3 Sec. 3.14) *"*Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system*."*

B. Software Safety Defined

Software Safety [3] is defined as "The discipline of software assurance that is a systematic approach to identifying, analyzing, tracking, mitigating, and controlling software hazards and hazardous functions (data and commands) to ensure safe operation within a system".

C. Software Reliability Defined

Software reliability as a discipline of software assurance defines that "Software reliability is the probability that software will not cause the failure of a system for a specified time under specified conditions. The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered" (ANSI/IEEE Std 729-1983.1).

D. Software Quality Defined

Software quality [3] is defined as "Quality is the degree to which an object (entity) (e.g., process, product, or service) satisfies a specified set of attributes or requirements".

E. Software Quality Assurance

Software Quality Assurance [3] defined as "The function of software quality that assures that the Standards, Processes, and Procedures are appropriate for the project and are correctly implemented".

III. DIMENSIONS OF QUALITY

A quality framework looking at an eight dimension product quality developed by Garvin [4] and a five dimension model of service quality derived by Parasuraman [5] as shown in Table 1. The framework is design by taking these dimensions of quality into an account.

TABLE 1. DIMENSIONS OF QUALITY

Frame Work	*Dimension*	*Definition*
Product quality (Garvin (1987))	1.Performance	Master controlling characteristics
	2. Feature	Accessories to basic working characteristics.
	3.Reliability	Probability of product failure within particular time period.
	4.Conformance	Product meets constituted standards.
	5.Durability	A measure of product life.
	6.Serviceability	The fastness and easiness of repair.
	7.Aesthetics	How a product looks, feels, tastes and smells.
	8.Perceived Quality	Comprehended by a customer.
Service Quality (Parasuraman et al. (1991))	1.Tangibility	The quality of equipment being perceivable by touch and appearance.
	2.Reliability	Perform the predicted service constantly with no mistakes.
	3.Responsiveness	Temperament to look after the customers and give immediate service.
	4.Assurance	Freedom of doubt that the firm giving to its client to inquire trust and confidence.
	5.Empathy	Personalized attention that the service provider gives its client.

IV. QUALITY PROGRAM STRUCTURES IN LARGE PROJECTS

The *Handbook of Software Quality Assurance [6]* gives the organizational structure (shown in Fig. 1), which exists in larger organization producing engineering or scientific applications. This organization chart taken from the actual project.

The framework is design by Quality Program Structure for Larger Projects into an account.

APM- Assistant Project Manager.
PQM- Project Quality Manager.

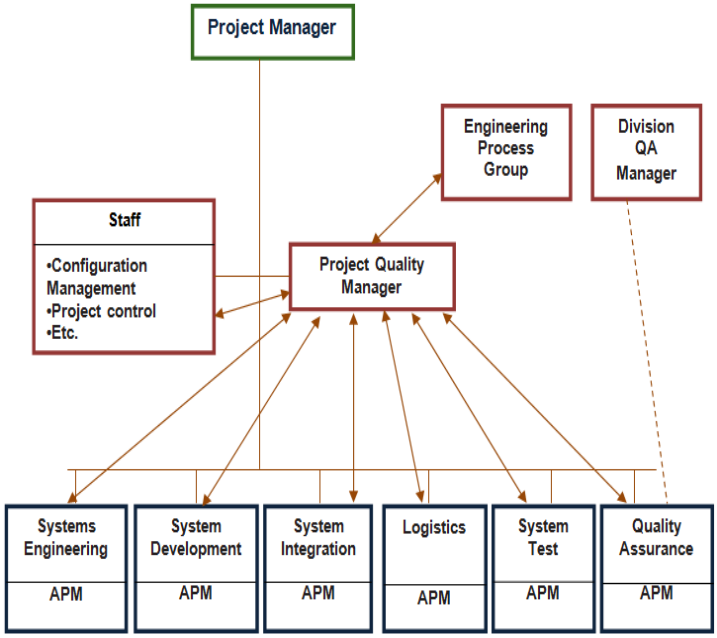

Figure 1: Structure for Larger Project Organization

V. SOFTWARE FAILURE LEVEL

Radio Technical Commission for Aeronautics (RTCA) safety critical working group *RTCA SC-167* and the European Organization for Civil Aviation Equipment EUROCAE WG-12 jointly prepared a standard called DO-178B for developing avionics software-intensive systems. The purpose of D0-178B [7] is "*to provide guidelines for the production of software for airborne systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements." DO-178B* defines levels of safety criticality. These are shown in Table 2:

TABLE 2: SOFTWARE FAILURE LEVEL

Level	Failure	Description
A	***Catastrophic***	*Fulminate and entire failure of system where recovery is unimaginable.*
B	***Hazardous***	*Failure has a large veto affect on performance or safety.*
C	***Major***	*Failure is substantial, but has a lesser impact than a Hazardous failure.*
D	***Minor***	*Failure is noticeable, but has a lesser impact than a Major failure.*
E	***No Effect***	*Failure has no affect on safety.*

VI. FRAMEWORK FOR RELIABILITY & QUALITY ASSURANCE PRACTICES APPLIED TO SCS

The framework is divided into different sections and description regarding each section is given below (Refer Page7, Figure 2 for each section):

6.1 Software Reliability

Reliability [8] is defined as a "*Characteristic of software, express by the probability that the item (software) will performed its required function under given condition for a time interval*". From qualitative point of view, reliability can be defined as "*the ability of an item to remain functional*".

6.1.1 Software Reliability Modeling

Various statistical models and techniques have been developed in software reliability engineering for forecasting and predicting reliability of software and number of remainder error in software [10].

Mainly software reliability models are of two types:

a) ***Deterministic:*** It examines the number of discrete operators and operands in code as well as the number of machine instructions and errors occur in the program.

Well-known models are: McCabe's cyclomatic complexity metric and Halstead's software metric.

b) ***Probabilistic:*** It represents the failure happenings and the fault removals as probabilistic events. It can be separated into different groups

- Error seeding
- Failure occurrence rate
- Curve fitting
- Markov structure
- Reliability growth
- Non homogeneous Poisson process.
- Time-series

6.1.2 Software Reliability Metrics

Failure occurrence expressions and data are use to derived Reliability metrics [9].

Generally used reliability metrics are-

a) ***ROCOF (Rate of Occurrence of Failures)*** related to the intensity of failure.
b) ***Availability*** is the probability that the system will be functioning at a committed time.
c) **MTTF** (***Mean Time to Failure)*** is the anticipation of the expecting time of the first failure.

6.1.3 Basic Mathematical Concepts

Reliability is one of the quality features that user require from the product manufacturer.

Mathematically, we called reliability $R(t)$ *as* the probability that a system will be successful in the time interval 0 to *t.* It is given as:

$$R(t) = P(T > t) \text{ Where } t \geq 0 \qquad (1)$$

Where T is a random variable denoting the time-to-failure or failure time. *Unreliability* $F(t)$, a measure of failure, is defined as the probability that the system will fail by time *t:*

$$F(t) = P(T \leq t) \text{ for } t \geq 0 \qquad (2)$$

$F(t)$ *is* the failure distribution function. If the time-to-failure random variable T has a density function *f(t),* then

$$R(t) = \int_{t}^{\infty} f(s)\, ds \qquad (3)$$

6.2 Software Quality Assurance (SQA)

6.2.1 Software Quality Assurance Policy & Objectives

Throughout the project lifecycle, Software Quality activities are conducted with the aim to provide the deep perception into the maturity level and quality of the software processes and related work products. Software Assurance Group (SAG) Software Quality personnel enforce work instructions, standard procedures, tools and techniques to objectively evaluate processes and work products, draw the inferences based on those evaluations, and providing effective communication to ensure resolution of all disobedience issues with managers and stakeholders.

Following are the objectives for SRQA [11]

a) Establish a common framework for the SRQA processes.
b) Setup and affirm the involvement of different groups involve in SRQA process.
c) Support and utilize the independent reporting structure.
d) Define SRQA activities, actions and jobs to assemble the goals of software assurance.

6.2.2 Procedures and Guidelines

For developing and implementing Software Quality programs Software Quality (SQ) personnel is responsible who provides applicable procedures, step-by-step instructions, and checklists for doing SQA process and product judgments throughout the life cycle of the software.

a) Procedure for Developing and Implementing Software Quality Programs.
b) Software Quality Activity Matrix: Activities of Software Quality that should be performed during each development phase (i.e., concept formation, requirements analysis, designing, development, integration, testing, user trails, operation and maintenance).
c) SQA Data Management Plan: A database should be setup for collection and storage of the different types of data related with the software quality activities and its artifacts.

6.2.3 Work Instructions

Work instructions are the standard procedures and instructions for performing Software Quality process and product assessments gradually throughout the software development life cycle.

a) Software Quality Assessment Process.
b) Software Quality Assurance Engineering Peer Review Assessment.
c) Software Quality Reporting Process.

6.2.4 CHECKLIST

Numbers of checklists are provided to aid Software Quality department in assessing procedures and products linked with software quality assurance.

6.2.4.1 Systems Review Checklists
6.2.4.2 Documentation Checklist
6.2.4.3 Other Assessment Checklists

6.2.5 Centralist Quality Repository Database (CQRD)

A centralized database containing information that can be accessed to produce regular reports and metrics. The database maintains the records regarding quality assurance procedure, process assessment checklists, findings, observations and standards that can be easily sharable.

6.2.6 Forms and Templates

The different SQA plans are developed in early stages of, and in parallel with project planning which defines the ways that will use for monitoring and assessing software development process and products. The templates are based on IEEE 730-2002 Standards:

a) Software Quality Assurance Plan
b) Software Quality Assessment Plan
c) Software Quality Assessment Report
d) SQE learning and Experience Log: documentation of SQ learning and work experience gained during project.
e) SQ Stakeholder role table: documentation of project stakeholder's role and involvement by software development life cycle phase.

6.2.7 Motivation & Training

Software Quality personnel should have basic idea in the following disciplines through prior experience, training, or certification in methodologies, processes, and standards.

a) Software Quality Assurance
b) Software Safety
c) Quality Audits and Reviews
d) Risk Management
e) Configuration Management
f) ISO 9001 and other Standards
g) IEEE Standard for quality and reliability assurance.
h) CMMI.

6.3 Safety

Software safety involves the systematic approach to determining, analyzing, and chasing software moderateness and to ensure software safety, control of hazards and hazardous functions (e.g., data and methods) within a system. Software is called *Safety-Critical* if it is fulfill at least **one** of the following criteria:

1. Make hazard or leads to a hazard.
2. Provides control for hazards.
3. Operates Safety-Critical methods.
4. Notice and report, or takes disciplinary action, if the system moves towards hazardous state.
5. Processes Safety-Critical data or commands.
6. Cause destruction if error occurs in the software.
7. Reside in a system as Safety-Critical software.

Software Safety program commences from Requirement phase and carries out throughout the software development life cycle including hazard analysis in each phase.

6.4 Software V&V

Software verification and validation (V&V) [13] procedure decides whether the development of each activity is exactly same as per the requirements throughout the product development and whether the software fulfills the user needs.

Objectives of Performing V&V are to-

1) Alleviate former detection and rectification of software bugs.
2) Deep perception of management risk involve into process and product development.
3) Ensure throughout the life cycle that the software development should be in specified schedule and budget.

V&V activities that are carrying out during the software development phases-

- User's requirements analysis and traceability.
- Design and code explanations and/or reviews.
- Conventional reviews.
- Software fault analysis.
- Documented test programs and process.
- Test planning, execution, and reporting.
- Audits and assessments.

6.5 Software IV&V

Independent Verification and Validation (IV&V) is carrying out by an establishment that is independent of the development organization in all respect and governing program/project management.

IV&V activities involves:

- Software design validation to match system requirements.
- Safety-Critical requirements traceability.
- Critical algorithm's design analysis.
- Coding analysis.

VII. CONCLUSION AND RECOMMENDATION

As every day, we are facing new challenges in software development for Safety-Critical Systems; SQA is a very complex and its ultimate goal is to deliver successful project. Also software safety and reliability playing key role in success of critical mission.

Software is a vital part of most systems. It controls hardware and provides mission-critical data. Software must be safe. Safety is not the sole responsibility of the System Safety engineer. Creating a safe system is a team effort and safety is everyone's responsibility.

The SQA applied from requirements analysis to software configuration to Critical Systems may be executed with the formation of a framework based on organization procedures and standards for SRQA of SCS which results the high-quality software.

ACKNOWLEDGMENT

The author is thankful to Professor Alka Gulati, Department of Computer Science & Engineering, for giving valuable suggestions and support on the paper.

REFERENCES

[1] Rodriguez-Dapena, "*Software Safety Certification: A Multinational Problem,*" *IEEE Software*, July/August 1999, p. 31, © 1999 IEEE.
[2] G. Gordon Schulmeyer "*Handbook of Software Quality Assurance*", Fourth Edition, ARTECH HOUSE, INC. 2008, pp. 212-213.
[3] The NASA Software Assurance Standard, NASA-STD-8739.8.
[4] Garvin A., "*Competing on the Eight dimensions of Quality*" Havard Business Review Nov-Dec 101–109, 1987.
[5] Parasuraman A., Berry L., Zeithaml V., "*Refinement and Reassessment of the SERQUAL scale*", Journal of Retailing 67(4), pp. 420–450, 1991.
[6] G. Gordon Schulmeyer "*Handbook of Software Quality Assurance*", Fourth Edition, ARTECH HOUSE, INC. 2008, pp. 28.
[7] DO-178B, "*Software Considerations in Airborne Systems and Equipment Certification*", RTCA publication.
[8] Reliability Engineering, Alessandro Birolini, Fourth Edition, pp 2-3.
[9] Reliability Engineering, Alessandro Birolini, Fourth Edition, pp 248.
[10] System software reliability, Hoang Pham, Series Edition, pp 153-176.
[11] Software safety standard, NASA Technical Standards, NASA-STD-8719.13B
[12] IEEE 730-2002 Standard for Software Quality Assurance Plans.
[13] IEEE Standard for Software Verification and Validation, IEEE Std 1012-1998.
[14] Software Quality Assurance, from theory to implementation, Daniel Galin, Pearson Education Limited 2004.
[15] Al-Qutaish R., "*Measuring the Software Product Quality during the Software Development Life-Cycle: An International Organization for Standardization Standards Perspective*", Journal of Computer Science 5 (5), pp. 392-397, 2009.
[16] ISO, International Organization for Standardization, "*ISO 9004:2000, Quality Management Systems-Guidelines for performance improvements*", 2000.

6. *APPROACH FOR RELIABILITY & QUALITY ASSURANCE OF SAFETY-CRITICAL SOFTWARE*

6.1 Software Reliability

6.1.1 Software Reliability Modelling
a) Deterministic
b) Probabilistic

6.1.2 Software Reliability Metrics

a) Static Code Metrics.
i) Line Count
1.1.1 LOC [1], 1.1.2 SLOC [2].
1.2 Complexity and Structure
1.2.1 CC [3], 1.2.2 Number of Modules, 1.2.3 GOTO [4].
1.3 Object-Oriented
1.3.1 Number of Classes, 1.3.2 WMC [5], 1.3.3 CBO [6], 1.3.4 RFC [7], 1.3.5 NOC [8], DIT [9]

b) Dynamic Metrics
1. Failure Rate Data, 2. Problem Reports

6.1.3 Basic Mathematical Concept

6.2 Software Quality Assurance (SQA)

6.2.1 Software Quality Policy & Objectives

6.2.2 Procedure & Guidelines
1. Procedure for Developing and Implementing Software Quality Programs, 2. Software Quality Activity Matrix, 3. Software Safety assurance body's SQA Data Management Plan

6.2.3 Work Instructions
1. Software Quality Assessment Process, 2. Software Quality Assurance Engineering Peer Review Assessment, 3. Software Quality Reporting Process

6.2.4 Checklist

6.2.4.1 Systems Review Checklists
1. Full System Concept Review, 2. Software Requirements inside system Review, 3. Preliminary Design Review, 4. Critical Design Review, 5. Test Readiness Review, 6. Software Quality Acceptance Review, 7. Operational Readiness Review, 8. System Operations Review, 9. Mission Operations Review

6.2.4.2 Documentation Checklists
1. Configuration Management Plan, 2. Risk Management Plan, 3. Software Development Folder, 4. Software Interface Control Document, 5. Software Maintenance Plan, 6. Software Management Plan, 7. Software Quality Assurance Plan, 8. Software Requirements Specification, 9. Software Requirements Traceability Matrix, 10. Software Test Plan, 11. Software Test Report, 12. Software User's Guide, 13. Version Description Document

6.2.4.3 Other Assessment Checklists
1. Code Inspection, 2. Engineering Peer Review, 3. Functional Configuration Audit, 4. Lessons Learned, 5. Measurement & Analysis Process, 6. PPQA External Assessment, 7. Project Monitoring, 8. Project Planning, 9. Requirements Management, 10. Software Configuration Management, 11. Software Problem Reporting, 12. Software Risk Management Process, 13. Software Unit Test Process

6.2.5 Centralist Quality Repository Database (CQRD)

6.2.6 Forms & Templates
1. Software Quality Assurance Plan, 2. Software Quality Assessment Plan, 3. Software Quality Assessment Report, 4. SQE Learning and Experience Log, 5. SQ Stakeholder role table

6.2.7 Motivation & Training
1. Software Quality Assurance, 2. Audits and Reviews, 3. Risk Management, 4. Configuration Management, 5. ISO 9001, 6. Software Safety, 7. Contracts/Contractor Surveillance, 8. CMMI.

6.3 Safety | **6.4 V&V** | **6.5 IV&V**

Acronyms
1. Lines of code 2. Source lines of code, 3. Cyclomatic Complexity, 4. Number of Go To Statements, 5. Weighted Methods per Class, 6. Coupling Between Objects, 7. Response for a Class, 8. Number of Child Classes, 9. Depth of Inheritance Tree.

Figure 2: A Framework for Reliability & Quality Assurance of Safety-Critical Software