A cooperative approach to ensure software product quality

Sangeetha.M[1], Arumugam.C[2], Sapna P.G[3] and Senthil Kumar .K.M[4]

[1] Coimbatore Institute of Technology
Coimbatore, India.

[2] Coimbatore Institute of Technology
Coimbatore, India.

[3] Coimbatore Institute of Technology
Coimbatore, India.

[4] Kumaraguru College of Technology
Coimbatore, India.

Abstract

Quality is the process of ensuring that software developed satisfies customer requirements. A mong the many software quality attributes like functionality, usability, capability, maintainability, etc., reliability is a major factor to assure quality of the software. Reliability ensures that software is failure free. I n this work, we propose to ensure quality through a behavioral model that evaluates business requirements and gives priority for quality attributes. The model consists of behavioural and human perspectives in assessment. It is used for assessment of software developed.

Keywords *– Quality, Reliability, Metrics, software assessment, human perspective.*

I. Introduction

The size and complexity of software systems have increased manifold. Also, software permeates every aspect of human life today. In this regard, there is need to ensure high quality of developed software. I EEE defines software quality as the degree to which a s ystem, component or process meets customer requirements [1]. S oftware requirements specified by the customer are recorded in the Software Requirements Specification (SRS) document. O ne way to ensure quality is by ensuring reliability of developed software.

Software pervades almost all aspects of everyday life today. In this regard, the quality of software plays a very important role. The focus of quality principles are: 1. Software should meet customer requirements. Lack of compliance with customer needs indicates poor quality software. 2. There is need to establish standards for defining good quality software[10]. D irect and indirect measures are used for the same. D irect measures include Lines of Code(LOC) and Function Points whereas indirect measures to assess quality include factors like usability, maintainability and reliability.

ANSI defines software reliability as the probability of failure-free software operation for a specific period of time in a s pecified environment [3]. C urrent techniques in software reliability research use failure data during integration and system testing phases to assess reliability. However, data collected in the testing phase does not indicate problems with reliability

early in the software development life cycle thereby leading to high costs and lack of time to take rectification action.

Companies could not provide any justification on the quality of their products to the users and users are left with uncertainties on the standard and quality of the software [4].There are other issues that relate to quality attributes, among them are priorities and different views of quality among users, developers and managers. It is recognized that views of users, developers and managers are different. A manager is more interested in the overall quality rather than a specific quality characteristic thus requires assigning weight to reflect the business requirements [5].

Section 2 surveys literature available in the area of software reliability engineering. Section 3 explains the proposed approach to ensure quality of the product. Case study is used for studying the applicability of the approach, discussed in section 4. Section 5 concludes the paper and discusses areas for future work.

2. Literature Survey

Software quality can be distinguished in two ways:

- External Quality
- Internal Quality

Bad external quality occurs because of system crashes, unexpected behaviour, data corruption, slow performance etc.

Internal quality is the hidden part and contributes to external quality. T his includes aspects like program structure, coding practices, maintainability, and domain expertise. External quality is a symptom whereas the root problem is internal quality. Poor internal quality leads to high maintenance costs. In order to improve software quality, internal quality must be improved.

2.1. The FURPS model (1987)

Hewlett-Packard developed a s et of software quality factors that make up its name, FURPS[2]. This model takes five characteristics of quality attributes -Functionality, Usability, Reliability, Performance and Supportability. When this model is used, two steps are considered: setting priorities and defining quality attributes that can be measured [6]. One disadvantage of this model is that it does not take into account the software product's portability [7].

2.2. ISO 9126(1991)

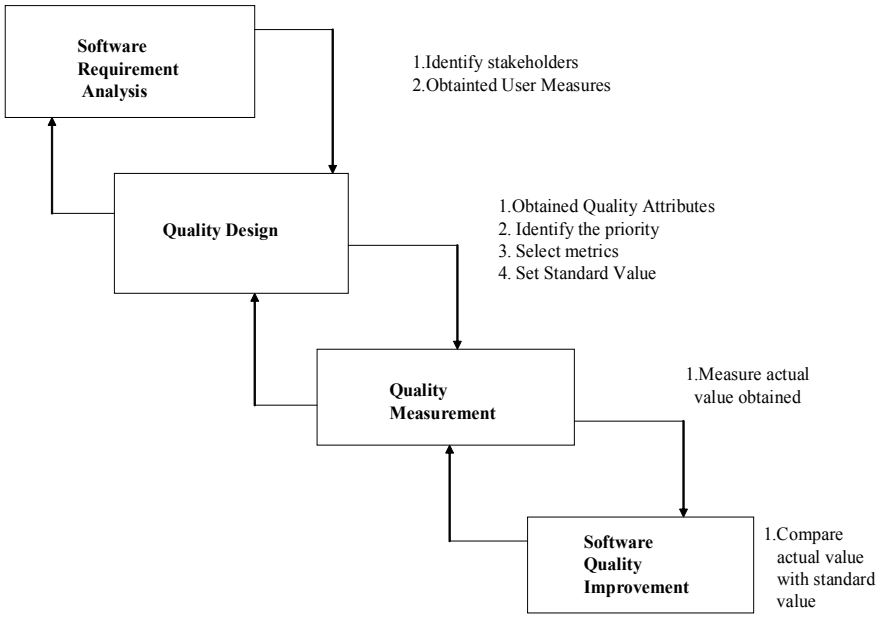
Many researches done investigated software assessment and quality using the ISO/IEC 9126 model as their guidelines in the assessment. Characteristics defined in this model are functionality, efficiency, usability, maintainability and portability. Disadvantage of this model is not show clearly how these aspects can be measured.

Literature on software quality shows that there are a number of characteristics that contribute to the behavioral aspects of quality. A classification of characteristics might be necessary to group characteristics according to their importance. Our research proposes an attribute weight and classification to be included in the quality model. This work will discuss in the following section.

3. Proposed Approach

Measuring Quality:

The nature of some internal attributes is that, despite being categorized as internal, they affect or influence external measures. Reliability is measured externally by observing the number of failures in a given period of execution time during a trial of the software, and internally by inspecting the detailed specifications and source code to assess the level of fault tolerance.


Software Requirement Analysis
1.Identify stakeholders
2.Obtainted User Measures
Quality Design
1.Obtained Quality Attributes
2. Identify the priority
3. Select metrics
4. Set Standard Value
Quality Measurement
1.Measure actual value obtained
Software Quality Improvement
1.Compare actual value with standard value

3.1. ISO 9126 attributes

The ISO 9126 a ttributes describe the external quality characteristic of specific software and its behaviour in different environments. This includes Efficiency(f1), Functionality(f2), Maintainability(f3), Portability(f4), Reliability(f5), Integrity(f6) and Usability(f7). These quality attributes are decomposed into several sub attributes to associate with direct measurable metrics.

Values for metrics range from 1 to 10.

$Q_m = f(f1,f2,f3,f4,f5,f6,f7)+€$ (1)

Where € is error term.

3.2. Technical measurement of human factor

This factor focuses on human aspect of quality towards the product. It helps to check the conformity of software to the user needs. This attribute is decomposed into two sub attributes.

- Userperception(p)={popularity, performance, law & regulation, recommendation, expectation, adaptability}
- UserRequirements(r)={acceptable, satisfactory}

Therefore,

$Q_h=f(p,r) + €$ (2)

3.3. Responsibility and Measuring Metrics

It defines the responsibility of a person to answer for the metrics related questions. Here, we calculate thWe used Likert scale of 1 to 5 based on team members. Likert scale posses set of attribute statements. It asked to express agreed or disagreed of a five-point scale.

Each degree of agreement is given a n umeric value from 1 to 5.

Recommendation Scale:

- 1= unacceptable
- 2= below average
- 3= average
- 4= good
- 5= excellent

$$TP=\sum_{i=1}^{n} P \quad (3)$$

Where P = priority of each external quality factor defined in section 3.1, TP = total priority of all factors, n=number of attributes defined in the analysis.

Calculate weight:

Weight $(W_i)=(P/TP)$, (4)

and

$\%W= (P/TP)*100$ (5)

where subscript 'a' represents an attribute.

After analyses, function point approach is used to group and classify attributes into three.

- Low
- Moderate

Levels	Attribute	Weight Factor
Low	Flexibility Portability Interoperability	1-4
Moderate	Safety Efficiency Maintainability Usability	5-7
High	Functionality Reliability Integrity	8-10

- Hi
  gh

Table 1: Example of the score obtained by behavioural attributes

Table 1 s hows score obtained by behavioural attributes defined in [9] which we adopt for our work.

Another aspect of quality which deals with human aspect is shown in the following equation:-

$$Q_{h=w_j USp + w_k USr} \quad (6)$$

Usability attribute is one of the best factor to balance between technical and human aspects.

Total quality of the software is measured using this equation:

$$Q_p = Q_m + Q_h \quad (7)$$

This model represents quality of software based on behavioural and human perspective.

4. Case Study

As initial study, the approach proposed in this work has been applied to medium sized project. First, requirements of the product were elicited. Further, users (customers) of the system provided input for the human factor depicting quality requirements of the user. Developers and managers provided external quality attributes of specific product.

The initial study reveals that the approach considers quality with reference to both functional as well as customers need thereby assuring the quality of the software.

As usual with case studies, a number of threats affect the validity of the results and demand for replication of the study, so as to confirm or confute our findings. Therefore, further investigation needs to be done on real-time projects to replicate the results done in the initial study.

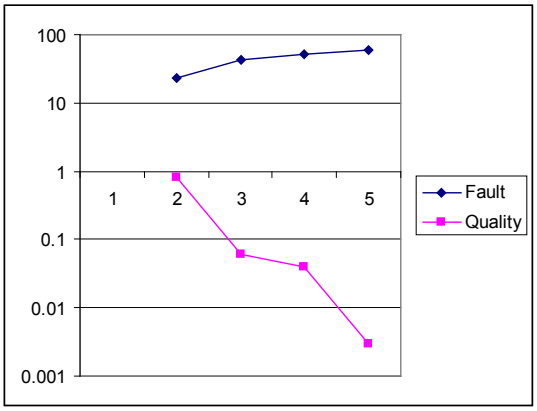


Figure 3 Behavior faults increases Quality will decrease

Figure 3 shows number of faults increases because of improper behavior of user ,quality in terms of reliability will decreases.

5. Conclusion and Future Work

The model defined in this work provides an explicit process for ensuring that software built is of quality. For this, both ISO attributes as well as human factors are considered thereby establishing a link between the technical as well as human inputs. This way, user preference and inputs is considered assuring that the product satisfies the customer.

An advantage of this model is that it considers external characteristics of quality (i.e. behaviour of software with relation to its environment) and gives added weightage to the human aspect. However, the model may be further refined to suit specific software requirements.

References

[1] ANSI/IEEE, Standard Glossary of Software Engineering Terminology, STD-729-1991, ANSI/IEEE, 1991.

[2] J.D. Musa, Software Reliability Engineering: More Reliable Software Faster and Cheaper (2nd Edition), AuthorHouse, 2004.

[3] Deepak Pengoria, VIT University Saurabh Kumar, A Study on Software Reliability Engineering Present Paradigms and its Future Considerations, 20th International Symposium on Software Reliability Engineering, 2009.

[4] J.A. Whittaker, & J.M. Voas,, "50 years of software:Key principles for quality," *IEEE IT Pro* (Nov/Dec), pp.28-35, 2002.

[5] M. Svahnberg, C. Wohlin, L. Lundberg & M. Mattsson, "A method for understanding quality attributes in software architecture structures," *ACM*, 2002.

[6] K. Khosravi, & Y.G. Gueheneuc, "A quality model for design patterns," Summer 2004".

[7] ISO/IEC 9126, "Software quality characteristics and metrics-Part2: External metrics", Technical Report, ISO/IEC JTC1/SC7/WG6, 1996.

[8]ISO/IEC9126-1: 2001,"Software engineering-product quality-part1:
quality model", 2001.

[9] Jamaiah Haji Yahaya, Aziz Deraman and Abdul Razak Hamdan, Software Quality from Behavioural and Human Perspectives, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.8, August 2008.

[10] R.A. Khan, K. Mustafa, S.I, Ahson, "Software Quality: Concepts and Practices," Alpha Science, Oxford, U.K, 2006.

M.Sangeetha received her M.Tech in Information Technology and her M.B.A in General.She is pursuing Ph.D in Computer Science Engineering. Currently, she is a L ecturer of Computer Science Engineering and Information Technology Department, Coimbatore Institute of Technology, Coimbatore.Her area of interests includes, Software Quality, Software Testing, Software Requirement Specification, Resource Management Technique and Information Security.