Software Quality: A mosaic of abstractions

Manpreet Singh Lehal
Assistant Professor
Lyallpur Khalsa College, Jalandhar

1. ABSTRACT

Today software market is highly volatile and competitive. The complexity of commercial software is on the rise along with the time constraint to produce high quality software. Increased software complexity probably leads to more and more defects but still organizations can afford it at the cost of being initiators. The idea of this paper is to nuance and provide an overview of the landscape of what sometimes briefly (and mostly thoughtlessly) simply is labeled quality. Quality can be a very elusive concept that can be approached from a number of perspective dependent on once take and interest. Every organization is unique and has its own needs so it is difficult to recommend “one size fit all” solution.

KEYWORDS Software Quality, SQA, Quality model, ISO, CMM

2. ISSUES AND DIFFICULTIES RELATING TO QUALITY

Software quality is concerned with how well the software meets the technical requirements contained in the specifications, emphasizing the conformance to specification. Quality is easy to recognise but hard to define and impossible to measure [1]. Quality is a multi-dimensional construct and has been classified according to a number of views. The notion of “quality” is not as simple as it may seem. For any engineered product, there are many desired qualities relevant to a particular project [2]. The software quality being a subjective term leads to different understandings and interpretations. So the perception of the term is different for different people. It depends on many factors like who is the customer and what is their influence on the things relating to the software. There may be n number of customers to a single software development project like the customer contract officers, management, end-users, customer acceptance testers, accountants, developer, salesperson, future software maintenance engineers, etc. Each customer will have his own definition of “quality”. Some might define it in terms of usability and some in terms of cost effectiveness.

David Garvin studied how quality is perceived in various domains; including philosophy, economics, marketing, and operations management [3]. He concluded that "quality is a complex and multifaceted concept" that can be described from five different perspectives:

The transcendental view (quality can be recognized but not defined) - Quality can be recognized through experience and not in tractable forms. A product of quality will stand out and is easily recognised. It refers to the excellence or elegance. It cannot be applied in a meaningful sense to a large software project as it would be constrained by resources.

The user view (fitness for purpose) - The view proposed by Juran in 1940s is highly personalized and concerns the extent to which a product meets the needs and expectations of the users. It relates to attributes like usability, reliability and efficiency. However, it is difficult to ensure that the software solution addresses the right problem. Users do have the ideas of what they want but they may express them in ways that designers could hardly comprehend. The designer would attempt to turn user’s ideas into requirement specification and then a design. This view will be determined by the quality of the match between the requirements and the design.

The manufacturing view (conformance to specification) - The manufacturer’s notion is to reduce the cost of development and maintenance. The CMM and ISO 9001 models are based on this view. This is the most attractive as it’s the easiest to quantify. Its guiding principle is the zero defect approach. It maintains that errors are avoidable but expected. But being error free is not the only standard for high quality software. Moreover its hard to convince people that zero defect state is really achievable.

The product view (quality as tied to inherent characteristics of the product) - This approach assumes that good internal product properties will result in improved external product behaviour. It implies, the higher the quality, the higher the cost. But others may believe that improvement in quality at the manufacturing stage lead to less defects and greater reliability thus reducing the cost of wastage and maintenance.

The value-based view (quality is dependent on the amount a customer is willing to pay for it) - This is the ability to provide the customers’ requirements at an affordable price. The cost here covers the resources rather than mere financial cost. People, time and tools may act as constraints upon the attainment of desired level of quality.

These diverse views can be complemented to some extent. If the user explicitly states his requirements specification, the technical specification can be derived directly from it. The problem arises when there is a conflict in the requirement of the different customers. The user may focus on the usefulness of the product while the manufacturer's priority may be on minimizing rework. Here the value based view of quality comes into effect which can effectively manage conflicts involving requirements change.

On the whole quality to a large extent is determined by people. Its people, who define problems, specify solutions, implement designs, produce and test codes and finally make judgement about the quality. Other tools are aids to enhance quality provided that the people are motivated towards their effective use.

3. MODELLING QUALITY

Quality can be depicted in a hierarchical way with the use of models. Various researchers have produced models of software quality characteristics or attributes that are helpful for discussing, planning, and rating the quality of software products. The models often include methods to “measure” the degree of each quality attribute the product attains.

McCall's model is one of the earliest models for quality assurance. It is aimed to be used during the development process [4]. It identifies three areas of software work (Figure I)

1. Product operation requires that it can be learnt easily, operated efficiently and results are in accordance with the needs of the user.

2. Product revision relates to error correction and adaptation.

3. Product transition is likely to increase the importance of the product.

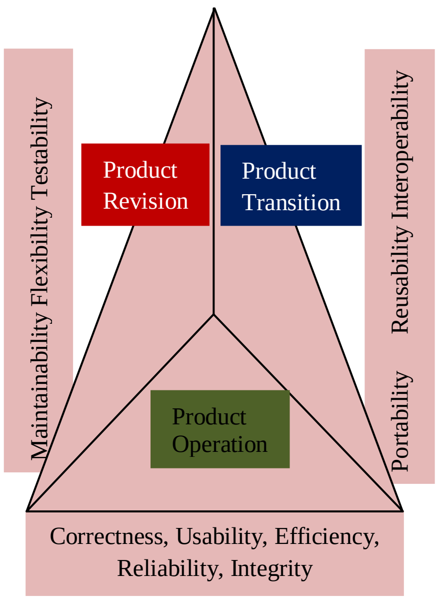
Maintainability Flexibility Testability
Product Revision
Product Transition
Reusability Interoperability
Portability
Product Operation
Correctness, Usability, Efficiency, Reliability, Integrity

Figure I: McCall's triangular quality model

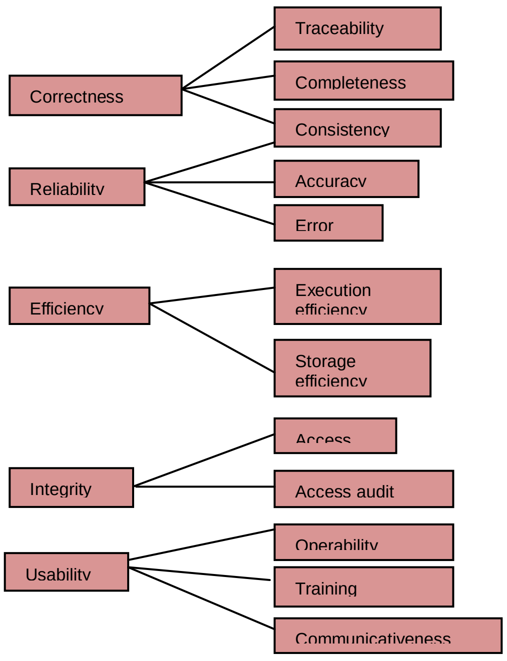

Figure II: Hierarchy of 11 quality factors (on the left hand side of the figure) related to 23 quality criteria (on the right hand side of the figure).

This model defines software-product qualities as a hierarchy of factors, criteria, and metrics. The arrows indicate which factors the criteria influence [5, 10]. A quality factor represents a behavioral characteristic of the system. A quality criterion is an attribute of a quality factor that is related to software production and design. A quality metric is a measure that captures some aspect of a quality criterion. Thus, the 11 quality factors contribute to a complete picture of software quality (Figure II & III).

One or more quality metric should be associated with each criterion. Thus, as the Figure III shows, you can measure portability by combining self-descriptiveness, modularity, software system independence, and machine independence.

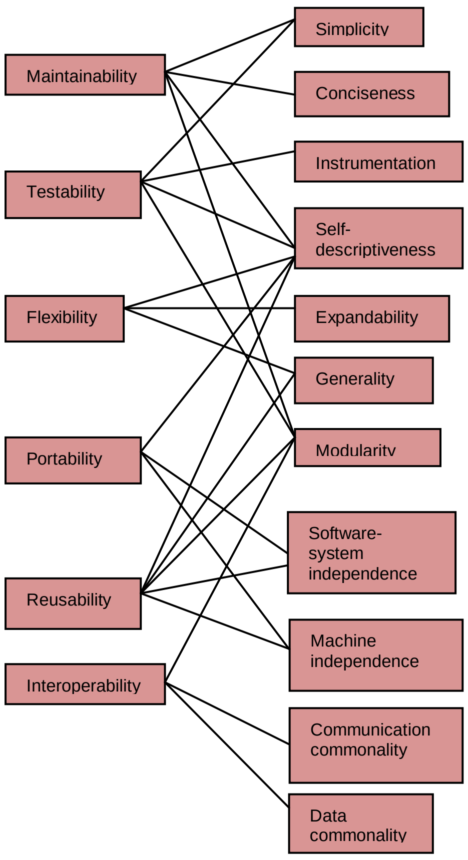

Figure III: Hierarchy of 11 quality factors (on the left hand side of the figure) related to 23 quality criteria (on the right hand side of the figure).

The metrics are derived from the number of "yes" responses to questions whose answers are subjective, such as "Is all documentation structured and written clearly and simply such that procedures, functions, algorithms, and so forth can easily be understood?" Dividing the number of yes responses by the number of questions gives a series of values in the range 0 to 1. The measures can be composed into either measures of specific factor quality or the product's quality as a whole by considering the relevant selection of questions. However, there are problems with values derived in this way. The degree of subjectivity varies substantially from one question to another, even though all responses are treated equally. This variation makes combining metrics difficult, if not impossible. Moreover, when appropriate, response complexity should be reflected in a richer measurement scale. For example, while it is reasonable to expect a yes-or-no response to the question, "Does this module have a single entry and exit point?" questions about documentation clarity probably require a multiple-point ordinal scale to reflect the variety of possible answers.

4. ISO CERTIFICATION

The standard of quality certification is set by ISO which gives an assurance to the customer that the organization is capable of delivering quality software product.ISO 9000 is the name of a series of QA standards recognized by over 200 countries around the world and adopted as their national standards for QA [6]. An important document in the ISO 9000 series is ISO 9001.The audits are actually carried out against this code (two other documents, ISO 9002 and ISO 9003, can also be audited against but these are 'cut down' versions of ISO 9001 for companies which do not need to comply with the entire code). The other documents in the ISO 9000 series list explanations to apply ISO 9001. It has 20 clauses that lay down guidelines which define the essential features of the software quality management system and suggest controls and methods that will allow the software to meet customer needs.

The principal features of ISO 9001 are that it,

- takes the basic principle of QA (the need for documented systems to support QA), and adds requirements to control system documentation to make sure it is kept up-to-date
- requires companies to carry out their own internal audits of their QA system to make sure it is working properly
- requires the QA system to be constantly monitored to ensure that it is effective, and that changes are constantly made to improve it

5. SQA AND QUALITY

The quality of software can be adjudged in totality if we complement engineering aspects with management aspects.SQA is a management system devised to assure quality.
Software Quality Assurance (SQA) is a systematic approach which helps to evaluate quality on the basis of its adherence to standards and procedures. SQA makes it sure that standards are established and followed throughout the life cycle of software acquisition. Process monitoring, product evaluation, and audits evaluate the compliance with agreed-upon standards. Software development and control processes should include quality assurance approval points, where an SQA evaluation of the product may be done in relation to the applicable standards [7]. SQA aims to establish a quality process, and suitably befits in the software life-cycle (SLC). A question arises about what should be the attributes of quality, and the answer is everything [8]. Obviously, this should be tailored to fit the circumstances, but full-blown SQA recognizes that it must interact with every aspect of the SLC to some degree. For example, the SQA team might analyse the work products produced at each stage against the desired product characteristics specified in the planning phase. Also, SQA commonly administers tests at both the code level and in verifying requirements fulfillment. Brown describes a number of specific activities that should be evaluated by the SQA team as below [9].

Software Requirements

-software development plan
-interface requirements specification
-software management quality plan
-software requirements specification

Preliminary Design

-top level design document
-test plan
-user manual

Detailed Design

-interface design document
-unit test cases
-integration test cases
-test descriptions

Coding and unit test

-source code
-unit test procedures and results
-integration test procedures and results

Integration and testing

-all revised program plans
-all revised description documents
-test procedures

System testing

-test report
-revised source code
-version description document

Again, this is but a small part of the full list of items that may be evaluated by SQA.

The first and foremost requirement in SQA is that it is a separate group responsible for quality in an organization. They set the goals, standards and mechanisms. The group assist the software development team in managing the quality requirements. Every software has some quality goals specified by the customers. These quality goals are to be achieved by the development team by introducing a set of activities or ensuring the delivery of quality to the customer. SQA activities operate on the normal activities of quality management. These activities play the role of monitoring, tracking, evaluations, auditing and reviews to ensure that the quality policy of the organization is implemented. SQA has a variety of tools to implement the policy which are auditing, inspection, technical review, co-ordinate and controls, collect and analyse and statistical quality control.

6. SQA & CMM

SQA assures that activities of software Configuration Management (CM) are in compliance with the standards and procedures laid down by CM. SQA reviews the plans and provides follow-up for nonconformance in case there are gaps in the requirements and implementations of policies. SQA prepares a report after auditing the CM functions for adherence to standards and procedures. The CM activities monitored and audited by SQA include baseline control, configuration identification, configuration control, configuration status accounting, and configuration authentication. SQA also monitors and audits the software library. Software development libraries provide for proper handling of software code, documentation, media, and related data in their various forms and versions from the time of their initial approval or acceptance until they have been incorporated into the final media. Approved changes to base lined software are made properly and consistently in all products, and no unauthorized changes are made.

7. SQA AND V&V

SQA assures Verification and Validation (V&V) activities by monitoring technical reviews, inspections, and walkthroughs. The role of SQA is to verify, participate when required and observe that the inspections are conducted properly. SQA also ensures that any actions required are assigned, documented, scheduled, and updated. Formal software reviews are done from time to time to identify problems and to find out whether the product meets all requirements. Preliminary Design Review (PDR), Critical Design Review (CDR), and Test Readiness Review (TRR) are some of the formal reviews. A review observes the overall phase of the product development to determine if the requirements are satisfactorily fulfilled. Reviews are part of the development process, designed to provide a ready/not-ready decision to begin the next phase. Decisions are reached by comparing the actual work with the established standards.

The objective of SQA is to assure that the Management and Development Plans have been followed, and that the product is ready to proceed with the next phase of development. SQA can only advise the management and it is up to the management to accept it or not.

8. CONCLUSION

So, we have seen that it is difficult to define software quality in totality. There is no single method to produce it and measure it. Quality is inhibited by the complexity of the system, invisibility of the system, sensitivity to tiny mistakes, entropy due to development, interaction with external components, and interaction with the individual user. In summary, while we may strive hard to produce the best possible quality, the customers may not even notice the difference between the best possible quality and pretty good quality. Quality is a mosaic of abstractions that emerges from the observed, the observer, and from the process of observation itself.

REFERENCE

[1] B.A.Kitchenham,Beverley Littlewood, Measurements for Software Control and Assurance' Springer, 1989.

[2] Dolores Wallace, Larry Reeker, 'Software Quality', Swebok, 2001

[3] D. Garvin, "What Does "Product Quality" Really Mean?" Sloan Management Review, Fall 1984, pp. 25-45.

[4] Software Quality Models and Philosophies, www.bth.se/com/besq.nsf/

[5] Gillies, Alan, 'Software Quality, Theory and Management', Thomson, 2006.

[6] ISO 9001 Quality Systems - Model for Quality Assurance in Design/ Development, Production, Installation, and Servicing, International Organization for Standardization, Geneva, 1994.

[7] Software Quality Assurance, Concepts and definitions, www.forum.onestoptesting.com/

[8] M. Evans and J. Marciniak, Software Quality Assurance and Management, John Wiley & Sons, New York, 1987.

[9] Du, Gengshen, Chad La Fournie, Software quality Assurance

[10] Waman, S Jawadekar, 'Software Engineering , Principles and Practice', Tata McGraw,2007.