A New Adaptable Quality Model for Software Quality Assurance

G. Vijaya[1], S. Arumugam[2]

Abstract

The model allows the specification of benchmarks against which achieved quality levels can be measured and provides guidance for building quality into software. The feasibility of quality goals is controlled by the use of a Relationship Chart and a Polarity Profile. Moreover, the ADEQUATE approach is not static; if project personnel changes occur, or project requirements change, the Relationship Charts and Polarity Profiles can be updated to reflect these changes. The aim of this work is to present a new model for software quality assurance which addresses the problems of these approaches and includes quality factors that represent a common set of criteria while allowing tailoring to a local environment. In addition the proposed model allows the quality factors to be determined and analyzed in an integrated, adaptable fashion.

Key words: Quality Modeling, Quality Measurement, Quality Assurance, Quality Factor

1. Introduction

A metrics measurement-based framework, linked to a quality model, is a requirement for effective software production and quality [2]. A key feature of the approach is Consistency; the method can be applied not only to Products, but also to the Processes and Resources used to create that Product. Thus, it is possible to model a whole organization which is defined by the Products, Processes and Resources it uses and creates. This enables the approach to form an overall quality assurance framework [1]. Many of the early designs of quality models have followed a hierarchical approach in which a set of factors that affect quality are defined, with little scope for expansion. Early efforts include the Factor Criteria Metric Model [3, 4] and the Hierarchical Quality Model [5]. Problems with these models include bias towards maintainability and reliability, non-independent factors and little recognition of different quality requirements for different projects [3, 6, and 7]. In order to address these issues, models were developed that follow a Define Your Own Approach [8], in which a collective decision is made between the developers and the users, as to what attributes constitute quality.

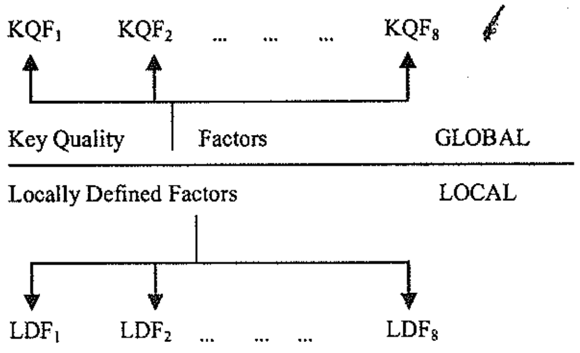
The main view is to present a new approach to quality modeling which seeks to combine these modeling approaches, whilst resolving conflicts of opinions of quality, so that quality measurement can be both tailored to a local environment and potentially can be compared across projects. This approach forms the basis of the Adaptable Quality Model, or ADEQUATE for short [1].

2. The Adequate Quality Modelling Approach

The following figure shows the pictorial representation of the ADEQUATE approach. The quality of the 'explicit and/or implicit attributes' forms a set of Key Quality Factors and a set of Locally Defined Factors.

[1] Assistant Professor, KGiSL Institute of Technology, Coimbatore. Email : dd.doca@kgisliim.ac.in

[2] Professor and CEO, Nandha Engineering College, Erode.



The key quality factors and locally defined quality factors:

Totally there are eight KQFs defined. Those are

- Maintainability
- Usability
- Cost/Benefit
- Security
- Reliability
- Timeliness
- Correctness
- Customer's View

Maintainability : In software testing, based on the definition given in ISO 9126, it is the ease with which a software product can be modified in order to:

- Correct defects
- Meet new requirements
- Make future maintenance easier, or
- Cope with a changed environment.

Maintainability is ability of product to be modified. It is included in the KQF primarily due to its perceived importance in other models [3, 5, 9-11].

Usability : Usability is defined as the ability of a product to be used for the purpose chosen.

Cost/Benefit : Cost/Benefit is defined as the ability of a product to satisfy its cost/benefit specification. The Costs and Benefits involved in a product's creation should be a major consideration [12]. If the costs are high and the benefits of its development are low, then there is little point in developing the product.

Security: Security is the condition of being protected against danger, loss, and criminals. In the general sense, security is a concept similar to safety.

Security has to be compared and contrasted with other related concepts: Safety, Continuity, and Reliability. The key difference between security and reliability is that security must take into account the actions of active malicious agents attempting to cause destruction.

Reliability : In general, reliability is the ability of a person or system to perform and maintain its functions in routine circumstances, as well as hostile or unexpected circumstances. And also the ability of a product to reproduce its function over a period of time and is also included in other approaches [3, 9-11].

Timeliness : The characteristic of a measurement reported in real time or shortly after the events it measures. Timeliness is defined as the ability of a product to meet delivery deadlines.

Correctness : * Conformity to fact or truth.

- The quality of conformity to social expectations

The ability of a product to meet and support its functional objectives. Other models also include this factor [3, 5, 9-11, 13]. If software doesn't meet its objectives, it may be delivered on time, but no one will use it.

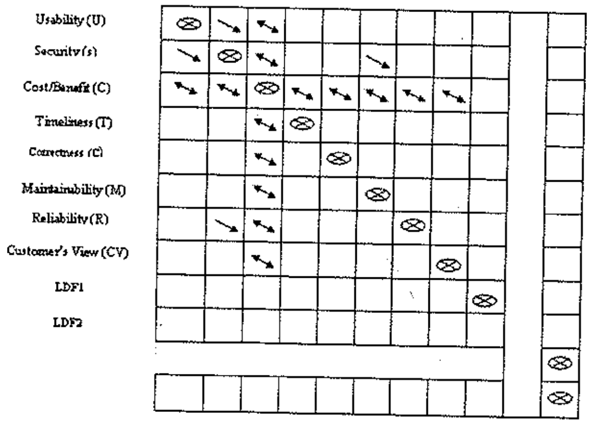
Customer's View : The customer's view of the value the company provides is based on the quality of the combined package of product and service [16]. This factor is to understand the opinion about the product from the customer. According to the customer's view the product is good or bad that can be found using this factor.

In fact the ISO/IEC 9126 model at the highest level uses six factors [14]. Based on the previous research [15], the number of key factors should be kept between three and eight.

The Relationship Chart

In this chart, each criterion is listed horizontally and vertically. Where one criterion crosses another, the relationship between those criteria is specified. For KQFs, the relationships are fixed; they are standard. This is because the set of KQF criteria does not change, therefore the relationships between each KQF criterion does not change. Across different projects, however, different LDFs may be used. The Relationship Chart (Fig. 2), therefore, shows the relationships between each LDF criterion and each KQF criterion and also the relationships between each LDF criterion and each other LDF criterion. Two quality criterias are given for the understanding purpose. They are Criterion A, Criterion B. The possible relationships between these two criteria are as follows:

1. **Neutral**: An improvement to the quality of Criterion A is unlikely to affect the quality of Criterion B.
2. **Direct**: An improvement to the quality of Criterion A is likely to cause an improvement to the quality of Criterion B.
3. **Inverse**: An improvement to the quality of Criterion A is likely to cause degradation to the quality of Criterion B.



Key:

↘ = Direct (If criterion A is enhanced, then criterion B is likely to be enhanced)

□ = Neutral (If criterion A is enhanced, then criterion B is unlikely to be enhanced)

↘ = Inverse (If criterion A is enhanced, then criterion B is likely to be degraded)

Figure 2: The Relationship Chart

The Polarity Profile

This is the second step to view the quality and also set the required goals for the each creation based on Relationship chart. Fig.3 shows an example Polarity Profile used in the ADEQUATE approach, it is clear the weight of each factor is set according to a particular requirement. As can be seen, both Maintainability and Security have been over-engineered, since their actual quality values exceed their required quality values (The criteria Efficiency and Portability are LDFs that have been chosen by the Essential Views). The criteria listed in the Polarity Profile are the same criteria as listed in the Relationship Chart.

There is a need, therefore, for Conversion Mechanisms which convert the results of metrics used to measure the quality of a criterion, into a value that lies in the range 1 to 5, for displaying in the ADEQUATE Polarity Profile. For each relationship type in the Relationship Chart (i.e., Inverse, Direct and Neutral), a set of rules exist which directly set the allowable required quality values. Consider two quality criteria, Criterion A and Criterion B. Given these two criteria, the

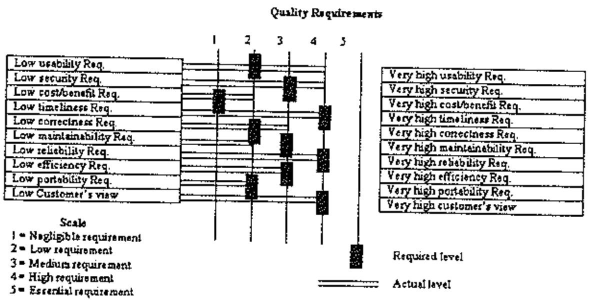


Figure 3: An example Polarity Profile

allowable required quality values in the Polarity Profile are determined using the following rules for a given relationship type specified in the Relationship Chart:

1) **Neutral:** No rules.

2) **Direct:** If Criterion A is greater than, or equal to the value 2, and then Criterion B must be greater or equal to the value 2.

3) **Inverse:** Neither Criterion A nor Criterion B can be set to the value 3. If Criterion A is set to the value 4 or the value 5, then Criterion B cannot be set to a value greater than 2. If Criterion A is set to the value 1 or the value 2, then Criterion B cannot be set to a value smaller than 4. Only the required quality values are constrained by these rules.

The Quality Formulas

Having considered both the Relationship Chart and the Polarity Profile it might be useful to produce a single value of quality which may be used to indicate the overall quality of a product in terms of its required versus actual values. Given a Polarity Profile showing both required and actual quality values, Fig. 4 shows the formulas used to produce this single quality value. Note that, where a criterion has been over-engineered, the required quality value is used in place of the actual quality value when calculating the overall actual quality score.

Measure	Formula
Required KQF	$RKQF = \sum_{i=1}^{i=n} KQFi$
Required LDF	$RLDF = \sum_{i=1}^{i=n} LDFi$
Actual KQF	$AKQF = \sum_{i=1}^{i=n} KQFi$
Actual LDF	$ALDF = \sum_{i=1}^{i=n} LDFi$
Overall required quality	$R = RKQF + RLDF$
Overall actual quality	$A = AKQF + ALDF$
Overall quality	$Q = (AX\ 100)/R$

Figure 4: Calculation Of The Overall Quality Value

3. Experimental Results

A software tool has been built to support the ADEQUATE approach which allows users to

- Create and view Relationship Charts
- Polarity Profiles and Overall Quality scores for a Product
- Process or Resource

The tool also provides a Definitions Database, allowing storage and retrieval of criterion definitions. A Measurements Database is also included, allowing storing and retrieval of ADEQUATE measurements. The software tool is implemented in Pentium IV 2.4 GHz using the Java language, tested under Microsoft Windows Environment.

The ADEQUATE approach has also been tested on a real project at a multi-national software institution named as KG Information Systems Limited. The above fig (3) shows the updated Polarity Profile screen from the ADEQUATE tool. As in Fig. (3), The Usability factor had the required quality value of 2, but the actual level is 4. According to this Software the usability factor is too good. Second level factor is Security. This is a web based application software. So the security level must be good. Here the required level is 3. But the actual level is 4. So here the Quality of the security is too good. The third level of factor is Cost/Benefit. Here in this factor the required level is 2 and the actual level is 3. So the Cost/Benefit is too supportable to the users. Timeliness is the fourth factor. The required level and also the actual are the same. The Timeliness met the actual quality requirements. Since the transfer of the handoff file no longer was delayed by being transferred through another external system. In the fifth factor Correctness, the required rate is 2 and the actual rate is 3. This mentions the software is good. Maintainability is the next factor and the required rate is 3, but the actual rate is not 2. So the maintainability is not too good. The next factor is Reliability. In this factor the Reliability required level and actual level set to 4. The required and actual reliability level is achieved well. In the Efficiency level 3 is required and also the Actual level. So the Efficiency is perfect in this level. Then another factor is Customer's view. Here the required level and the actual level set to 4. This result gives the good opinion from customers. This means according to the customer's view, this software is perfect. The overall Quality score achieved is 92%. Applying the ADEQUATE approach to this project has shown that the technique clearly captured the quality requirements; quality delivered and effectively highlighted differences between different implementations of the project. The approach allowed the building of a matrix of individual software quality factors with clear visibility and individual weighting according to project requirements.

The results have also highlighted and confirmed the impact of the relationships between the various quality factors, as well as the need to balance the requirements as well as the choices that have to be made. Essential views are considered and evaluated which lead to improvement in both individual software quality factors as well as the overall quality of the project.

4. Conclusion

This paper, proposed the new Quality measurement and Quality assurance approach. Quality factors are divided into two factors (KQF and LDF). Using this model we can measure the quality levels and provide the guidance for building quality into the software. Use of Relationship Chart and a Polarity Profile controlled the feasibility of quality goals. The KQF set consists of eight attributes. Eighth attribute is the newly added one, which is Customer's view. Already we saw about seven factors, Maintainability, Usability, Cost/Benefit, Security, Reliability, Timeliness, and Correctness. These seven attributes gave good results for testing the Software Quality. But newly added eighth attribute gave excellent results in this field. Use of this Customer's view attribute is really a good one. It is hoped that the next level of the software should be too excellent using these attributes. For further development of the system more statistical analysis can be done in the global and local factors and more attributes can be added.

Refrences

[1] Souheil Khaddaj and Gerard Horgan, *"A Proposed Adaptable Quality Model for Software Quality Assurance"*, Journal of Computer Sciences 1 (4): 482-487, 2005.

[2] Offen. R and R. Jeffery, *"Establishing software measurement"*, IEEE Software, PP. 45-3, 1997.

[3] Gillies.A.C, *"Software quality: Theory and management"*, Chapman and Hall, PP. 1-33, 1992.

[4] Kitchenham. B and S.L.Pfleeger, *"Software quality: The elusive target"*, IEEE, Software, PP. 21, 1996.

[5] Boehm. B.W, J.R Brown, H. Kaspar, M. Lipow, G. MacLeod and M.J Merritt, *"Characteristics of software quality"*, North Holland, 1978.

[6] Manns. T and M. Coleman, *"Software quality assurance"*, MacMillan Press Ltd., PP. 1-12, 1996.

[7] Matsumoto. Y and Y. Ohno, *"Japanese perspectives in software engineering"*, Addison-Wesley, PP. 279-30, 1989.

[8] Fenton. N, *"Software metrics: A rigorous approach"*, Chapman and Hall, PP. 42-62, 1991.

[9] Dromey. R.G, *"A model for software product quality"*, IEEE Transactions on Software Eng., 21: 146-162, 1995.

10] Plant. R.T, *"Factors in software quality for knowledge based systems"*, Information and Software Technol., 33: 527-536, 1991.

[11] Kitchenham. B, *"Towards a constructive quality model"*, Software Eng. J, PP. 105-113, 1987.

[12] Elbert. C, *" The road to maturity: Navigating between craft and science"*, IEEE Software, PP. 77- 82, 1997.

[13] Lindland. O.I, G. Sindre and A. Solvberg, *"Understanding quality in conceptual modeling"*, IEEE Software, PP. 42-49, 1994.

[14] ISO/IEC TR 9126: Software Engineering-Product quality, 19-12-2000.

[15] Miyoshi. T and M. Azuma, *"An empirical study of evaluating software development environment quality"*, IEEE Transactions on Software Eng., 19: 425-435, 1993.

[16] Don A Kennard, *"Reflecting the Customers' View"*-http://www.isixsigma.com/library/content/c020218a.asp

Author's Biography

G. Vijaya, received B.E.,(ECE) from PSNA College of Engineering and Technology in 1989 and receive M.E.,(CSE) from Kumaraguru College of Engineering and Technology. Currently doing Ph.D., in Anna University, Chennai. She is working as an assistant professor in the department of Information Technology in KGiSL Institute of Technology, Coimbatore. She has industry and academic experience. She is the member of IEEE, CSI and ISTE. Her main interest area is Software Engineering.

Dr. S. Arumugam completed his Bachelors degree in Electronics & Communication Engineering and M.Sc(Engg)., in Applied Electronics, both from PSG College of Technology under University of Madras and Ph.D., in Computer Science & Engineering from Anna University. He has been serving in the Directorate of Technical Education since 1974 onwards. He is also serving as a member of various boards of studies. He is a vice president of Computer Society of India for the following regions, Tamilnadu, Pondicherry, Andaman and Nicobar, Kerala, Lakshadweep. He is a senior member of IE, IETE and ISTE. His area of interest is Image Processing. He has so far published over 60 papers in various International, National Journals and Conferences.