Examining Stakeholder Requirements for Software Quality

Maria Haigh
University of Wisconsin,
Milwaukee, WI
mhaigh@uwm.edu

June Verner
National ICT Australia Ltd
Sydney, Australia
june.verner@nicta.com.au

Abstract

'High quality' might seem an obvious requirement for any piece of software, but do the different stakeholder groups involved in its production and use conceptualize this requirement in the same way? Many existing models refine the broad concept of quality into a number of well-defined and measurable attributes related to the software product itself and the development process which produced it. Until now, however, little attempt has been made to empirically examine the requirements for software quality held by different groups involved in the development process. We conducted a survey of more than 300 students and alumni of one of the leading Executive MBA programs in the United States, asking them to rate the importance of each of 13 widely-cited attributes related to software quality. The results showed business role-related differences in some specific areas and agreement in many others. We also consider the implications of these results and their relevance to software requirements analysis.

Keywords: Software quality metrics, perceptions, priorities, software stakeholders, business need, requirements.

1. Introduction

In 1964, U. S. Supreme Court Justice Potter Stewart was faced with the need to define obscenity. Abandoning any attempt to define specific acts, depictions or measurable characteristics he instead noted that "I shall not today attempt further to define the kind of material I understand to be embraced… [b]ut I know it when I see it." This statement would accurately capture the attitude of many people towards software quality. We all think we know what it means, but most people have difficulties in defining it. As a result we can no more be sure that two different groups would view a piece of software as high quality than we could be sure that the citizens of San Francisco and Salt Lake City would uphold the same standards of obscenity. Both are in the eye of the beholder.

To overcome this problem, many models of software quality have been proposed, each of which has tried to separate the broad concept of quality into a number of well-defined and measurable attributes related to the software product, its fidelity to requirements, and the development process which produced it. The best of this research, seeking empirical confirmation, has tied observed attributes to project outcomes [14].

Any software project includes several different sets of "stakeholders," including users and developers, and managers and non-managers. In this research, we conceive of these stakeholder responsibilities as being business roles adopted by particular individuals with respect to specific pieces of software. Someone with the stakeholder role of manager of development for one software project might be a user of another piece of software and a developer of a third. We see attitudes to software quality among these different groups as indicative of their perceptions of the business needs the software will be required to satisfy. In this sense, software quality requirements may be thought of as a specialized subset of business requirements, or at least as desired characteristics that will allow the software to satisfy those requirements.

Our research asks whether these different stakeholder groups value the same attributes when defining their requirements for software quality. By asking a variety of software stakeholders to evaluate the importance of different commonly used attributes of high quality software we aim to determine their implicit personal definitions of software quality. This allows us to explore the relationship between business roles and software requirements. If profound differences are found between holders of different stakeholder roles, this signals a need to take steps to bridge this cultural gulf between participants. Alignment of software quality conceptions between holders of these different business roles will allow organizations to devote resources to agreed upon high-priority attributes with an expectation that all stakeholders groups will value the results.

2. Background

Requirements for software quality can be defined from many points of view, depending on the role the person plays with the software and on the type of system being developed [1], [3], [6], [10]. Existing research shows that we have to view software quality requirements not as an absolute measure, but in terms of trade-offs [7]. The implications for requirements analysis and perceptions of business need are obvious. If quality is refined to a set of effective and comprehensible metrics, then the required and desired levels of each attribute can be specified during the requirements specification phase of any project [5], [9]. Because recent models indicate correlations (both negative and positive) between desirable attributes (such as maintainability and efficiency), devoting resources to maximizing inappropriate attributes might actually damage the effectiveness of the software produced [8]. Quality therefore can be viewed as a set of unavoidable trade-offs, existing beyond the familiar tensions between time, cost, and quality.

A better understanding of software quality requirements for different stakeholder groups will lead to better communication between the parties involved with the system. To understand business need, managers and developers should understand what aspects of software quality are important to them, and to users, so that they can ensure that developers of the system implement the features with the highest priority.

3. Method

We conducted an online survey of 315 software stakeholders. The survey made available using a web interface connected to a database. The URL was distributed via email to the Executive MBA students and alumni at one of the most highly ranked business schools in the United States. Distribution of the survey to this sample facilitated reaching a homogeneous group of people with the same education, yet representing managers, users, and technical personnel from all sectors of the U.S. economy.

Respondents used a wide variety of different software packages. We therefore asked each respondent to select the piece of software most important to them in carrying out their work responsibilities and answer questions with respect to this piece of software. This gives more meaningful results than simply asking the respondent about his or her attitudes to software in general.

Stakeholder role was defined with respect to the specific piece of software chosen for evaluation. We used two axes on which to divide our respondents into four distinct software stakeholder roles. There is an axis of users versus developers: stakeholders who are involved in managing or performing the software development process and those who are not directly involved in these tasks. There is also an axis of managerial versus non-managerial responsibilities with regard to the software.

We are interested in finding out whether members of the four different stakeholder groups largely agree on the priorities assigned to different software quality attributes or whether widespread and systematic divergences exist in the priorities assigned to different software quality attributes by members of the different stakeholder groups. Thus, the null hypothesis of the study can be expressed as follows:

> *H_0: There is no significant difference in software quality priorities between different software stakeholder groups.*

The corresponding alternative hypothesis is thus:

> *H_1: There is a significant difference in software quality priorities between different software stakeholder groups.*

The survey included questions covering stakeholder's job function, their relationship to software product most important for their job function, and a set of questions asking the respondent to rate the importance of each of 13 software quality attributes. Each attribute was rated independently on a scale of 1-7, where 7 meant very important and 1 meant not important.

The software quality attributes and accompanying definitions provided to the survey respondents were as follows.

- ACCURACY: The degree to which the software outputs are sufficiently precise to satisfy their intended use
- TESTABILITY: The effort required to test the software to ensure that it performs its intended functions
- USABILITY: The effort required to learn and operate this software
- SECURITY: The extent to which access to this software by unauthorized persons can be controlled

- EFFICIENCY: The amount of computing resources required by this software to perform its function
- CORRECTNESS: The extent to which this software satisfies its specifications and fulfills your mission objectives
- PORTABILITY: The effort required to transfer this software from one hardware configuration or software system environment to another
- AUGMENTABILITY (SCALABILITY): The extent to which this software can take advantage of additional resources to deal efficiently when increased demands are placed on it
- INTEROPERABILITY: The effort required to couple this software with another
- ROBUSTNESS: The degree to which this software continues to function in the presence if invalid inputs or stressful environmental conditions
- FLEXIBILITY: The effort required to modify this software for uses or environments other than those for which it was specifically designed
- MAINTAINABILITY: The effort required to locate and fix an error in this software, or to change or add capabilities
- REUSABILITY: The extent to which components or modules of this software can be used for other purposes

These attributes were selected from the review of existing literature [8]. The list attributes used is neither complete with respect to every attribute proposed in the literature, nor entirely orthogonal. Some of the attributes overlap in their meaning. Many of the attributes came from one of the most heavily cited software quality models - the Boehm et al. software quality model [2]. Boehm's model implies relationships between software quality attributes: the model is not a list of independent qualities, but an interconnected hierarchy of attributes. Some attributes from more recent quality models were incorporated, and many of the descriptions were updated or simplified to make them more relevant to non-specialists and to reflect technological changes.

4. Results

We present our results in the following order: a summary of the background of the respondents by industry sector, stakeholder, and application area of the software they evaluated. Our review of the results continues with a discussion of the data analysis.

4.1 Demographic and Related Data

The main purpose of the study is to explore the software quality priorities held by different software stakeholder groups. Each respondent identified him- or herself as either a user or developer of the software concerned, and as either a manager (managing its users or developers) or non-manager (personally using or developing the software concerned). Combining these two variables thus divided respondents into four groups, which we refer to here as stakeholder roles: User, Manager of Users, Developer, and Manager of Development. Table 1 shows the distribution of respondents by their stakeholder roles.

Table 1. Respondent distribution by stakeholder role

Stakeholder Group	Frequency	Percent
Developer	46	14.6
Manager of Development	52	16.2
User	155	49.2
Manager of Users	59	18.7
Missing Data	3	0.9
Total	315	100

Thirty one percent of the respondents were responsible for development of the software concerned: 16.2% were managing its development, while a further 14.6% were personally performing development tasks. The remaining 69% of the respondents were not associated with the development of the software evaluated, and are therefore treated here as users. Fifty percent personally used the software they evaluated and 18.7% identified themselves as managers of the users of the software they evaluated. (35% of the respondents fell into one or other of the management roles).

The respondents came from a variety of industries as shown in Table 2.

Table 2. Respondent distribution by industry sector

Industry Sector	Frequency	Percent
IT and Telecomm	92	29.2
Government	16	5.1
Healthcare	32	10.1
Manufacturing	55	17.5
Military	5	1.6
Academic and Research	15	4.8
Service-Non-Computer	100	31.7
Total	315	100.0

Most of the respondents (60%) came from two sectors: (1) IT and Telecommunications, and (2) non-IT services. Overall, however, seven major industry categories were represented.

Table 3 shows the distribution of stakeholder roles by industry. Responses associated with developers and developer managers mainly came from IT and Telecommunication industries: 43% and 44% respectively. The service-non-computer industry was the most represented for respondents not associated with software development: 39% of software users and 32% of user managers were from this industry. While each stakeholder role was found across the full range of industries, there is clearly some covariance between industry and role – some of which may reflect the nature of each industry and some of which may be due to random variation in the sample.

Table 3. Stakeholder roles by industry

Industry (column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
IT and Telecomm. n=92	43.4	44.2	21.3	25.4
Govt. n=16	10.9	1.9	3.4	6.8
Healthcare n=32	6.5	7.7	12.3	10.2
Manufact. n=55	13.1	13.5	18.7	22
Military n=5	2.2	3.9	0.7	1.7
Academic and Research n=15	6.5	11.5	3.2	1.7
Service-Non-Computer n=100	17.4	17.3	40	32.2

Respondents evaluated a variety of software packages. These packages were categorized across two axes:

- **software application area**: business administration, manufacturing or production, scientific/research activities, creativity-related software (e.g., games, art/graphics, music, etc.), and other;

- **software type:** off-the-shelf-software; off-the-shelf-software customized for respondent's company use, in-house developed software for sale, in-house developed software for the use within respondent's organization, and "other", software did not fit into any of the previous categories.

Table 4. Application areas of the evaluated software.

Application Area	Frequency	Percent
Business Administration	147	46.7
Creativity	4	1.3
Manufacturing	28	8.9
Other	100	31.7
Scientific	30	9.5
Missing values	6	1.9

Forty seven percent of the respondents evaluated business administration software, making this by far the most represented category of software in the survey. Thirty two percent of the software evaluated was categorized as "other" – meaning that the respondent did not believe it to fit into any of the pre-defined application area types. Scientific and manufacturing software were other two most popular application areas (9.5% and 8.9% respectively). (Table 4).

Table 5. Software application area chosen for evaluation by stakeholder role

Appl. Area (Column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
Business Admin. n=147	37.8	30.6	59.7	37.9
Creativity n=4	0.0	0.0	2.0	1.7
Manufact . n=28	8.9	24.5	2.0	15.5
Other n=100	44.4	24.5	28.6	37.9
Scientific n=30	8.9	20.4	7.8	6.9

Table 5 shows the software application areas evaluated by respondents in different stakeholder groups. Data in this table reflects missing data and rounding errors.

Table 6. Software type chosen for evaluation by stakeholder role

Software Type (Column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
Off-the-shelf-software	15.2	5.8	62.6	20.3
Off-the-Shelf-Customized	17.4	25.0	19.4	45.8
In-house developed to sell	39.1	32.7	7.1	8.5
In-house developed for the use within own organization	23.9	28.9	9.0	20.3
Other	4.4	7.7	1.9	5.1
Total	100	100	100	100.0

Table 6 shows the development sources of the software being evaluated by members of each stakeholder group. (Respondents were asked to evaluate the piece of software most important to them in carrying out their primary job functions). This shows that 62% of users primarily used off-the-shelf software for their business responsibilities. Developers and developer managers were involved with in-house software developed for sale, off-the-shelf customized software, and in-house developed software for internal use only. Business stakeholders along the managerial axis commonly used off-the-shelf customized software and in-house software developed for the use within their own organization.

Table 7. Average satisfaction with evaluated software by stakeholder groups

Stakeholdr Role	Satisfaction Avg
Dvlp.	3.78
Mgr. Dvlp.	3.88
User	3.95
Mgr. User	3.91

Respondents were reasonably happy with the software under consideration: 78.2% measured their satisfaction with the software as '4' on a 7-point scale.

The differences in software satisfaction between the stakeholder groups were not statistically significant. It is interesting to notice that both developer groups were less satisfied with software than either of the user groups. Developers and managers of development were thus more critical towards software than other stakeholders: they value software quality more and have higher expectations for the software products than respondents who are not involved with software development process.

In the next section we present the results of our analysis of the stakeholders' quality priorities regarding software used for their jobs.

4.2 Data Analysis Results

The aim of this research is to discover if there are systematic differences in software quality requirements priorities between respondents with different stakeholder roles associated with software. Individuals and, more importantly, stakeholder groups, showed substantial variance in the mean scores they assigned to attribute importance. This made the raw data less useful for evaluating systematic divergences in priorities. Our interest here is in software attribute priorities, which we operationalized as the importance assigned to an attribute by a given respondent relative to the average importance assigned by the same respondent to all attributes. These priority scores are obtained by applying simple linear scaling to the results of each respondent. Trochim [15] suggests this type of scaling: dividing the score assigned to an attribute by the sum of scores assigned to all attributes by the same respondent and then multiplying by the number of attributes (13 in our case). The formula for score scaling is as follows:

$$Adjusted\_Attibute\_Priority_{ij} = Raw\_Score_{ij} * N \; / \; \sum(Raw\_Score_i)$$

Where i is the record number (one record for each respondent); j is the column number (one column per each quality attribute); Raw_score is the rating entered by a respondent; N is the number of attributes, 13 in our case. Comparison of the importance of the software quality attributes mean frequency distribution analysis and ANOVA analysis were applied to examine collected data.

Differences in software quality attribute priorities between stakeholder groups revealed the following:

- Users ranked accuracy, correctness, integrity, interoperability, robustness, and usability higher than any other group.
- Developers ranked maintainability and testability higher than other groups.
- User managers ranked augmentability, efficiency, and flexibility higher than other stakeholders.
- For development managers reusability was more important than for other groups.
- Developers and development managers appear to be in general agreement. User managers seem to be closer in their software quality priorities to development managers (and to developers) than they are to users.
- Maintainability was significantly more important for managers and developers than for the user group.
- Testability was more important to the development managers and developers than the other stakeholder roles.

Table 8 shows rankings of all quality attributes within the different stakeholder groups. Software quality attributes in Table 8 are ordered by ranking for all respondents.

Table 8. Software quality attributes ranking by stakeholder role

Stake-holder Role	Dvl pr	Mgr. Dev	Usr	Mgr. User	All
**Correc.	1	1	1	2	1
Accuracy	2	2	2	1	2
**Usabil.	5	6	3	4	3
Robust.	3	4	4	3	4
Interop.	7	7	5	6	5
Integrity	8	8	6	7	6
**Maint.	4	3	8	5	7
Augment.	9	9	7	8	8
Effic.	10	10	9	9	9
**Testab.	6	5	11	10	10
Portabil.	13	13	10	12	11
**Flexib.	11	12	12	11	12
**Reusab.	12	11	13	13	13

The differences for testability and maintainability are not surprising: developers and development managers care more about these attributes because they are directly related to their responsibilities toward the software. Perceptions toward these attributes reflect their perceptions toward business need. These groups mainly dealt with in-house software developed for sale, off the shelf customized software, and in-house software developed for internal use only. They are the people responsible for developing or customizing the business software. Therefore, their perception of business need is to cut costs by developing software with the highest levels of maintainability and testability. They are concerned not just about the cost of developing the software but also for the long term cost of the software over its entire life. The results for other attributes raise questions of applicability to respondents' real experiences with software packages today. We can speculate on the inherent appeal of terms: "correctness", "accuracy", "integrity", "robustness" and their linguistic association with word "quality". Other terms such as maintainability, testability and reusability are less likely to be naturally associated with quality for those respondents who are without significant exposure to the specialized terminology of software development. This may explain why these attributes were the most important for the majority of respondents, and were ranked particularly highly by users – who as a group had little or no involvement with the software development process - certainly likely to be less than the other respondent groups.

Given the apparent agreement between users and developers on the general importance of attributes like "correctness" (very high for both groups) and "reusability" (low for both groups) we must, however, suggest that further research is needed to discover exactly why respondents ranked these attributes as they did. Such research should also investigate the results of modifying the supplied definitions, or using different but synonymous term (such as "Fidelity to Specification" rather than "Correctness").

Six software quality attributes showed statistically significant differences for the different stakeholder groups. The strongest results, and those that held up best under multivariate regression analysis, concerned three attributes: usability, testability and maintainability. While usability was ranked as one of the most important six attributes by members of all groups, users ranked it more highly than did the members of any other stakeholder roles. Importance of usability to users reflects their perception of business need. Users' business need consists of learning and using software, therefore, by definition, usability becomes very important. They are probably not interested in the software other than that it is easy to use and provides appropriate functionality.

5. Conclusions

This work explores the differences in software quality perceptions between different business software stakeholders. Three hundred and fifteen respondents ranked each of thirteen generally accepted attributes of software quality on a scale of one to seven according to their perceived importance for the piece of software most vital to that individual's work. We have identified that stakeholders required different types of software for their jobs and that majority of stakeholders in the non-development group are more satisfied with the software they are using.

The main conclusion of this study is somewhat surprising and positive in terms of its real-world implications: the null hypothesis has been largely upheld. Within this survey population few significant and systematic divergences were observed in the conceptions of software quality held between developers and users, and between managers and non-managers. Given widespread perceptions of fundamental cultural clashes between these groups, and equally widespread concern over the ability of software systems as implemented to satisfy real business needs, this is surely a reassuring finding.

Of course, the survey was administered to a group of respondents enrolled in or graduated from a leading executive MBA business school program. While the respondents filled a variety of stakeholder groups, they might reasonably be supposed to have been admitted into the program according to their managerial potential and to have been exposed to a demanding core curriculum and a strong shared culture during their studies. In this they are unlikely to be entirely representative of the broader population of users, managers and developers. Achieving such agreement in most organizations might require significant investments and the development of a strong cross-functional culture.

Within these constraints, our research suggests that a piece of software might plausibly satisfy the quality requirements of users, managers, and developers. One implication of this finding is that tactics such as formally specifying the required levels of each attribute early in the development process might win agreement across roles [5]. In particular, developers and developer managers were in agreement on software attribute priorities.

The survey did reveal significant differences between the priorities assigned to a number of attributes by holders of different roles according to their perceptions of business need: usability (favored by users) and testability and maintainability (favored by development staff). This suggests that attempts to educate users and developers about each others' priorities should be focused on these three attributes. For example, users might lack an appreciation of the relationship between testability and the other attributes with which they are more directly concerned. Fortunately, the attributes are not among those widely seen as hard to achieve in combination and so it may be possible to satisfy all groups (in contrast with the negative relationships sometimes identified between attributes such as flexibility and efficiency) [12], [14]. Armed with the knowledge of these systematic differences in perceptions, project managers may also be better able to deal with and balance the necessary tradeoffs.

6. References

[1] Arthur, l. J. Measuring Programmer Quality, John Wiley and Sons New York 1985

[2] Boehm, B.W., Brown, J.R., and Lipow, M. (Eds.), "Quantitative Evaluation of Software Quality", Proceedings of the Second International Conference of Software Engineering, (1976), pp. 592-605.

[3] Boehm, B.W., In, H. (1996), "Identifying Quality-Requirement Conflicts", IEEE Software, vol.13, no.2, pp.25-35

[4] Cerpa N. and Verner J.M., "Practitioners' Views of Software Quality", The International Journal of Computing and Engineering Management,, Vol. 4, No. 3, September-December 1996, pp. 1-15.

[5] Dekkers N., "Maximising Customer Satisfaction.", Proceedings of the 12th European Software Control and Metrics Conference in London Eds K. Maxwell, S Oligny, R Kusters and E. van Veenendaal, April 2-4th, 2001

[6] Deutsch, M. S and Willis, R. R. "Software Quality Engineering" Prentice-Hall Englewood Cliffs NJ (1988).

[7] Gentleman, W. M. "Software Quality World-wide: What Are the Practices in a Changing Environment", Proceeding of the Sixth International Conference on Software Quality (6ICSQ), Ottawa, Canada, 28-30 October, 1996, pp.335-345.

[8] Haigh, M "Software Quality Revisited: Diverging Priorities Between Stakeholder Groups?", Ph.D. Dissertation, Drexel University, 2002

[9] Jacobs, S. "Introducing Measurable Quality Requirements: A Case Study", IEEE International Symposium on Requirements Engineering June 07 - 11, 1999 Limerick, Ireland.

[10] Kusters, R.J.; van Solingen, R.; Trienekens, J.J.M., "User-perceptions of embedded software quality", Eighth IEEE International Workshop on Software Technology and Engineering Practice incorporating Computer Aided Software Engineering, p.184-97, 1997.

[11] McConnell, Steve. Code Complete: A Practical Handbook of Software Construction. Redmond, WA: Microsoft Press, 1993

[12] Perry W.E. "Quality Assurance for Information Systems: Methods, Tools and Techniques", QED Technical Publishing Group, 1991.

[13] Sheehan, G. M. (1955). An Application to Payroll. In R. N. Anthony (Ed.), Automatic Data Processing Conference (pp. 145-157): Graduate School of Business Administration, Harvard University

[14] Shumskas, A. F. "Software Risk Mitigation" in Schulmeyer, G. Gordon and James I. McManus, ed.. Total Quality Management for Software. 1992: NY, Van Nostrand Reinhold. pp. 190-220.

[15] Trochim, W. (2000). "The Research Methods Knowledge Base", 2nd Edition. Atomic Dog Publishing, Cincinnati, OH.

[16] Verner, J.M., Moores, T.T. and Barrett, R., "Software Quality: Perceptions and Practices in Hong Kong", in Achieving Quality in Software (eds) S. Bologna and G. Bucci Chapman and Hall 1996, pp. 77-88.

[17] Wallmuller, E., Software Quality Assurance: A Practical Approach, Prentice-Hall Englewood Cliffs NJ 1994

[18] Wilson, D. N. and Hall, Tracy, "Perceptions of Software Quality: A Pilot Study". Software Quality Journal, 7 pp. 67-75, 1998.