K. Henningsson and C. Wohlin, "Understanding the Relations between Software Quality Attributes - A Survey Approach", Proceedings 12th International Conference for Software Quality, Ottawa, Canada, Proceedings on CD, October 2002.

UNDERSTANDING THE RELATIONS BETWEEN SOFTWARE QUALITY ATTRIBUTES – A SURVEY APPROACH

Kennet Henningsson
Claes Wohlin
Department of Software Engineering and Computer Science
Blekinge Institute of Technology
PO Box 520
SE-37225 RONNEBY, Sweden

ABSTRACT

Software quality is a complex concept containing a large number of quality attributes. These attributes often have different meaning for different people and different attributes are not of equal importance. Moreover, the actual relations between the attributes are mostly poorly understood. Companies have to cope with these relations in their daily software development. On the one hand, companies take management decisions based on experience. On the other hand, researchers address software quality too. However, the two views are not necessarily the same.

To increase the understanding of software quality attributes and their relations, two surveys have been conducted. The first survey focuses on the research literature and the second is an interview survey with people from industry. From these surveys, it is concluded that there is an agreement, in qualitative terms, that quality attributes are dependent. However, different opinions exist about the actual relations. No quantitative relations have been found. The main conclusion is that there is a gap between research literature that poses mostly generic relations between quality attributes and the tacit knowledge in industry. The tacit knowledge within industry is largely focused on system specific relations between quality attributes. The result from these surveys provides a compilation of relations between quality attributes that illustrates the gap between the views in industry and academia respectively. The understanding of the gap is the first step towards bringing the two views closer to each other.

Introduction

Requirements on software products cover different aspects of the software including both functional and non-functional requirements. It is by no means an easy task to fulfill all requirements simultaneously. The requirements come from several directions and stakeholders of the final product. As a developing organization, project manager, or department manger it is difficult to manage the potential conflicting requirements and to handle the trade-offs between different requirements. This is particularly obvious when addressing the requirements related to different quality attributes. It means that it is necessary to prioritize between the quality attributes and even between different stakeholders' interests.

From a research context, different quality attributes have been addressed in numerous research papers, but very few papers address more than one or a couple of attributes. Exceptions do exist, and to mention a few. One example is the structure discussed in (McCall, 1994), where different quality attributes and their relations are addressed. In the context of the Win-Win method the prioritization of different quality, attributes are presented in (Boehm, 1996). Chung et al. take the most rigorous approach in (Chung, 2000), where a framework is described. The framework is complete with both a methodology and a notation for handling relations between quality attributes.

It is well known in industry that relations exist between software quality attributes. However, the knowledge is primarily tacit, meaning that the relationships are rarely explicitly stated, at least among the companies participating in our study. Moreover, the relations are not stated clearly within academia either. This alone poses a problem. In other words, it is not clear which knowledge is needed for making informed decisions and a correct prioritization to achieve the right software quality.

The objective here is present two surveys with respect to quality attributes, the potential conflicts between them and the trade-offs. The first survey is a literature survey to capture the understanding of the quality attributes in the research community. The second survey is an interview survey to capture the industrial practice and understanding of the quality attributes in an industrial context. The result is a listening of the existing relations among quality attributes, both from literature and from an industry perspective. These relations are intended to increase the awareness of potential problems due to relations between quality attributes.

The paper is structured as follows. Hypotheses and the research method are discussed in the next section, followed by the results from both the literature and industrial surveys are presented. The following section addresses the analysis of the result and some conclusions from the analysis. In the Influences on Quality section the impact on overall quality is addressed. The next two sections contain a discussion and some future work, which is followed by the conclusions.

Hypotheses and Research Method

As a starting point for the surveys, several hypotheses were stated. They are formulated to obtain a better understanding for the potential conflicts and the trade-offs needed between software quality attributes.

The following hypotheses are addressed using the surveys:

1. Relations between quality attributes exist.

2. There are conflicting relations between quality attributes.

3. There are supporting relations between quality attributes.

4. There is a lack of knowledge regarding different attributes' effect on each other.

5. Informed trade-offs are necessary in an industrial context.

The hypotheses are primarily formulated from an industrial perspective. The literature review of the state-of-the-art is used as a support to confirm or refute the hypotheses.

The survey method is used in the study (Babbie, 1990). The study includes a survey of both industry and research findings. The latter was carried out through a literature survey covering published material, including conference papers, journals, and books. The sample was mainly research databases.

When gathering information from industry, a structured interview approach was chosen. This method is chosen to facilitate clarification of the posed questions, as well as providing the interviewee with a wider set of answers, and not limiting the answers to a predefined set of answers (Robson, 1993).

The interviews were recorded and the answers were transcribed, though not with the exact wording. Eight interviews were carried out, but only seven of the interviewees were able to answer the questions in the way that was intended, i.e. one interview was not included in the answer set.

The sample for the industrial survey consists of five companies located in Blekinge, Sweden. The companies are all heavily involved in software development. Further, the companies are either competing on the international market or delivering their services to companies that are competing on the international market. At some of the larger companies, more than one individual was interviewed. The companies are labeled from A to E in the rest of the paper.

The participating individuals represent the local offices, and the number of employees within these local offices' varies between 15 and 150 people. The companies surveyed include both product developing companies as well as consulting companies, both internal consultants and external consultants.

Throughout the industry survey the same definitions were used, the definitions are stated by McCall (1994) and are given in Table 1, below. These definitions were made available for the interviewees during the interview.

Table 1. Quality attribute definitions stated by McCall (1994).

Name	*Description*
Correctness	Extent to which a program satisfies its specifications and fulfills the user's mission objectives.
Reliability	Extent to which a program can be expected to perform its intended function with required precision.
Efficiency	The amount of computing resources required by a program to perform a function.

Usability	Effort required to learn, operate, pre-pare input, and interpret output of a program.
Integrity	Extent to which access to software or data by unauthorized persons can be controlled.
Maintainability	Effort required locating and correcting an error in an operational program.
Flexibility	Effort required modifying an operational program.
Testability	Effort required to test a program to en-sure it performs its intended function.
Portability	Effort required transferring a program from one hardware configuration and/or software system environment to another.
Reusability	Extent to which a program can be used in other applications-related to the packaging and scope of the functions that programs perform.
Interoperability	Effort required coupling one system with another.

Result

This section includes first a general discussion of the results before addressing the results from the literature survey and the industrial survey separately.

The first hypothesis was quickly confirmed. There is a general agreement that different quality attributes are heavily dependent on each other, and hence that it is not possible to optimize all of them simultaneously. This became obvious from both the literature survey and the interviews with industry.

However, there were also varying opinions within industry, as well as in literature on what caused the relations. The opinions vary from that the relations are generic, i.e. the relations are present and the same independently of, for example, application domain, system type or the actual implementation. Other opinions raised claimed that the relations are system and implementation dependent. Thus, the relations are not depending on the quality attributes directly, but rather on the actual implementation. Some interviewees suggested that the relations between quality attributes are primarily governed through the implementation and in particular the code size of the software system. In the literature, McCall (McCall, 1994) and Chung et al. (Chung, 2000) provide suggestions of how the implementation affects the relations between different quality attributes. While others, for example Bosch (Bosch, 2000), are mainly focused on the architecture and how the architecture affects different quality attributes.

A reason for this difference may be the varying abstractions for which the differences are described. Some research material covers a fairly generic view on the relations, which means that the specific type of system or the application domain is not taken into account. Basically, this means that a more abstract and general view of the relations is assumed. On the other hand, one case study shows that some of the relations stated in other parts of the research community are not applicable in that specific case study, and that the general belief that performance and maintainability has a negative relation caused problems (Häggander, 1999).

This illustrates the need to enhance the understanding of the relations between software quality attributes, and also the need to being able to handle the trade-offs and conflicts between the attributes to obtain a suitable blend of qualities for the final software product.

The objective here is to provide a starting point to increase the understanding of software quality attributes. As a starting point, it was decided to survey both the research literature and the industrial view on the subject. Next, the findings from the literature are presented which then is followed by the industrial survey.

Literature Survey

This section describes the relations identified in the research literature. It is necessary to establish a common terminology and understand the problems with the accuracy of the relations stated in the research literature. This involves three issues, the first is definitions, the second is explicit knowledge, and the third is the abstraction level.

The first issue, definitions, is the fact that different definitions are used for the same name of the quality attribute. It is common that authors mention quality attributes by their name and not state the definition for the quality attributes. The definitions used in this report are the definitions found in (McCall, 1994).

The second issue, explicit knowledge, is based on the non-explicit information stated in research material. The alleged relations are not always clearly stated within literature. This causes problems of understanding relations as well as having confidence in the stated relations.

The third issue is the level of abstraction for the relations. It is rarely stated within literature on which abstraction level the relations are present. Examples are given where the stated relations are viewed as generic and applicable to all systems, e.g. (McCall, 1994). On the other end of the scale are studies of specific systems that are concerned with monitoring the relations between quality attributes. The latter provides local information, but very little is known in between local information and generic statements.

When describing the relations found in the research literature there are three categories used for labeling the relations: positive, negative, and no influence.

A positive relation means that the relating quality attributes are helping each other. This means that by increasing one quality attribute this will support the increase of other positively related quality attributes. A negative relation means that the relating quality attributes are conflicting. This means that while increasing one quality attribute, other negatively related quality attributes will be limited, or at least not as easily achieved. It is also stated in (McCall, 1994) that there are quality attributes that do not have any influence on each other.

A table containing the relations between quality attributes stated in literature can be found in (Henningsson, 2001). The table is a summary of relationships published in (Boehm, 1996, Kotonya, 1997, Bosch, 2000, McCall, 1994). It should be noted that the definitions are neither always clearly stated nor on which abstraction level the relations are present.

When examining relations in literature it is noted that the relation between assurance and usability are labeled as both positive and negative (Boehm, 1996). This depends on the actual realization used for achieving the quality attribute, assurance. This shows that the relations are stated with respect to methodologies for implementation and solutions within the final system. This is however not the case in other publications.

Relations within Industry

The relations found in the industrial survey are presented here. In order to avoid the problems with divergent definitions, the interviewees were presented with the definitions stated in (McCall, 1994). If the definitions given did not fit the interviewee's opinion of the quality to be described, the interviewee was allowed to make a new definition. This option was however never used.

The results from the industrial survey are shown in Table 2. The interviewees were not forced to address any specific attributes. They were allowed to determine themselves, which attributes to put on the list. This was done to also get a picture of which attributes the interviewees found particularly important.

The companies are given a code from A to E, to being able to separate the answers from one company to another.

Table 2. Relations found within the industrial sample.

Company	*Interviewee*	*QA*	*vs.*	*QA*	*Relation/Influence*
A	1	Flexibility		Efficiency	Negative
		Testability		Correctness	Positive
		Usability		Reliability	Positive
A	2	Time to Market		Correctness	Negative
		Time to Market		General Quality	Negative
B	3	Usability		Reliability	Positive
		Reliability		Maintainability	Negative
		Reliability		Correctness	Negative
		Usability		Efficiency	Negative
C	5	Time to Market		Maintainability	Negative
		Reliability		Maintainability	Negative
		Reliability		Efficiency	Negative
C	6	Correctness		Efficiency	
		Efficiency		Maintainability	
		Maintainability		Testability	
		Portability		Maintainability	
		Interoperability		Efficiency	
D	7	Correctness		Efficiency	Negative
		Usability		Efficiency	Negative
		Time to Market		Correctness	Negative
		Usability		Correctness	Negative
		Usability		Time to Market	Negative

E	8	Portability	Maintainability	Negative
		Availability	Maintainability	Negative
		Portability	Correctness	Negative
		Availability	Correctness	Negative
		Portability	Efficiency	Negative
		Maintainability	Correctness	Positive
		Availability	Efficiency	Negative

Interviewee number 6 was unwilling to state the actual relations. This is the reason that no information is provided in the right most column for the sixth interviewee. It is interesting to note that most relations noted are regarded as being negative. A further analysis of the relations is provided in the following section.

Analyses and Conclusions

Some commonalities in terms of viewpoints can be found from Table 2. The criterion for judging what is a commonality is simply that more than one interviewee has stated the relation. The common relations and the agreeing companies are listed in Table 3.

Table 3. Common relations found within industry

Quality attribute	*vs.*	*Quality attribute*	*Relation/Influence*	*Companies*
Usability		Reliability	Positive	A and C
Time To Market		Correctness	Negative	A and D
Reliability		Maintainability	Negative	B and C
Usability		Efficiency	Negative	B and D
Correctness		Efficiency	Negative	C and D
Portability		Maintainability	Negative	C and E

The relations listed show that there are some common opinions in industry. To be a common relation, more than one person has stated the relation. This indicates with some certainty that these relations are present within industry. A reflection is that the majority of the found relations are negative relations, only in one case industry agreed upon a positive influence between quality attributes.

Out of the six relations listed in Table 3, only two are found in the literature. The two are: Usability vs. Efficiency and Portability vs. Maintainability. For the first case, literature and industry agree that the relation is negative. In the second case, there is a disagreement. In the literature, it is stated that the relation is positive, but the industrial survey shows that they have the opposite opinion.

It is also clear, from looking at the relations stated in literature and industry, that there are more positive relations stated in the literature. A possible conclusion is that it is more important for industry to identify negative relations. This is probably dependent on more than one reason, but one contribution is believed to be that the industrial experience is mainly based on conflicts and consequences that originate from conflicting quality attributes. Further, there is in industry a tendency to not investigate why there are not any problems, if it works the question “Why?” is rarely asked, this can be a reason for why industry fails to state and find positive relations.

Influences on Quality

As stated above there are potential conflicts between quality attributes, and a conflict that is not handled in a proper way will, with high probability, cause problems. In this section, the influences that the relations might have on qualities of the software are discussed. Worth noticing is that no concrete solutions to the potential problems can be given. However, we are certain that the illustrations of these issues are helpful as such.

There are some basic understandings that need to be in place. First, quality is about perception. This means that, depending on your needs and how well the final system or construction can fulfill these needs, the grading of a particular quality attribute is influenced, where grading refers to the perceived importance of that attribute.

Second, different stakeholders have varying interest and hence prioritize quality attributes differently. The stakeholders also influence the requirements stated on the system, which will affect the first point. The success of the project is affected by the stakeholder acceptance or satisfaction (Kotonya, 1997).

The relations between quality attributes influence the overall quality of the final product. It has also been established that there are conflicts between quality attributes. The conflicts will make it, if not, impossible at least hard and expensive to achieve the required level for the involved quality attributes. In particular, it becomes obvious that it is not possible to optimize all of them simultaneously. This will lead to that some stakeholders may become disappointed, and thus are not willing to accept the product, which may lead to project failure.

There is also a risk for when prioritizing the quality attributes in the requirement process or later in the prioritization process that, the most current quality attributes will be prioritized. This can lead to prioritizing quality attributes that are conflicting with quality attributes that are equally important, but are visible later in the process, such as maintainability. For example, take the relation stated in (McCall, 1994) regarding maintainability vs. efficiency (negative), and maintainability vs. reliability (negative) that is found in the industrial survey. If both efficiency and reliability are prioritized early in the process, it is reasonable to believe that the maintainability for the product will suffer, and also that these limitations will only be detected later in the lifecycle of the system.

Another example of conflicting quality attributes and their consequences are the relation between Time To Market (TTM) and correctness, which is negative. As can be seen from Table 3, this was a relation stated by more than one industrial representative. The basic consequence is that when Time To Market for the project is shortened, and pressure is applied, the number of faults introduced is increased in combination with less testing effort.

There is also a risk for chain reactions. An example is constructed based on the relations in Table 3. Assume that a system is constructed with high demands on portability and correctness. According to the relations in Table 3, this will lead to low levels of maintainability and efficiency. In later stages of the lifecycle, higher efficiency is needed due to increased use of the system. It would be easier to adapt the system to this need if the system was easily maintained, which given the relations are not likely to be the case. This would probably lead to an expensive and problematic maintenance phase. Figures up to 60-80% of the total lifecycle cost may be spent on maintenance. An assumption is that if these relations and consequences are known and showed, it would influence prioritization, and it would be possible take an informed decision also reduce effort and cost and improve quality within industry.

Discussion

Trade-offs and conflicts between software quality attributes are difficult. There is no single solution taking care of all the issues involved. Two ways of solving the problem have been proposed in literature. One way is to work around the issue of conflicting quality attributes, this is the approach taken in (Häggander, 1999). The solution was to create an alternative implementation for the system that eliminated the relations between the quality attributes. This solution worked for the examined quality attributes, maintainability and performance, but the effect the alternative implementation had on other quality attributes is not mentioned. Moreover, it is not stated if the alternative implementation was actually used or not. This way of attacking the problem, also shows a conscious choice of which quality attributes that are of importance. A two-quality attributes approach may work as long as other quality attributes have much lower priorities than the two selected attributes. This is based on the assumption that the quality attributes that are not monitored were not prioritized.

Another approach is showed in (Chung, 2000) where the authors provide a framework for supporting the decision-making process, both for finding the quality goals and identifying and choosing the operationalizations (possible design alternatives) for meeting the quality goals for the target system. The NFR (Non-Functional Requirement) framework also visualizes the interdependencies between the quality goals and how the operationalizations support or limit one or more quality goals. This is a helpful way to actually get more information and knowledge about the relations between quality attributes.

In order to address the problem with conflicting quality attributes it is necessary to know their relations, if there is no understanding or knowledge about the relations, it is unlikely that a solution to the conflicts will be possible. The relations identified in this study provides some insights into what to look for within an organization, what are the alleged relations, and which are present in the current developing organization.

As the analysis of the result above indicates, a majority of the industrial representatives focuses on the negative relations. The reasons for this may vary, but some hypotheses are: industry is focusing on the negative relations based on the experience of conflicts among quality attributes, and industry has a product focus and is not able to see the effects the positive relations have on other quality attributes.

A first step in solving the problem is to acknowledge the problem, and raising the issue of relating quality attributes as well as start monitoring the quality in terms of quality attributes to be able to see the effects certain decisions will have. When an organization knows what relations that are present for their operation and which relations that are likely to have the most serious impact on overall quality, a great deal is won. A good way to start is to see which of the generic relations described in literature are present, and this will also provide guidance on finding the positive relations.

Finally, it is important to note that research is needed in the gap between generic relations and the relations found for a particular system. There is a need of knowing more for certain types of systems, certain applications and so forth. An increased understanding with respect to this gap is needed to further enhance our opportunities of developing software systems with predictable quality and a suitable trade-off between software quality attributes.

Future Work

All parties concerned with software engineering are responsible for the overall quality of the software. This concern and responsibility becomes even larger and more serious along with the increasing dependencies people have on computers and computer-based systems.

It is necessary to move from the tacit knowledge within industry and the non-explicit information in the research literature. If this is not done, relations between quality attributes and consequences thereof will still be hidden in a mist, and will only become apparent when they cause problems and negatively affect the overall quality of software systems.

A challenge for doing so is the lack of measurements or metrics connected with the quality attributes within industry. Measurements are the place to start to be able to monitor the effects of relationships between quality attributes. The responsibility for doing so is not only the responsibility of industry. Academia and the research community must provide industry with working tools and support in order to make the collection of metrics monitoring the relations between quality attributes possible.

The survey of industry gave the result that most of the measures were quantitative, but only a subset of the quality attributes were measured, and this subset consisted of course of the most prioritized quality attributes. This is also an obstacle in the quest of finding and establishing both positive and negative relations.

One way of doing so is to further investigate what is actually happening with the qualities of the software depending on which solution is chosen (Chung, 2000). This is referred to as the operationalizations and their effect on the quality goals. The industrial representatives expressed a need for this type of approach during the interviews. One opinion was that the relations between the software quality attributes are basically decided by the code size and the actual implementation. Thus, there is a need for understanding different solutions' influence on the quality attributes and their actual relations rather than just resorting to generic statements about the relations. Some research in this direction is being conducted (Svahnberg, 2001) although much more is needed.

Conclusion

The hypotheses stated in the second section are supported by the literature and industrial surveys. From the study, the following can be concluded with respect to the hypotheses:

1. It is clear from both the literature and the industrial surveys that relations do exist between many of the software quality attributes. There are however several unanswered questions with respect to how attributes are related and what drives a relation.

2. Conflicts between software quality attributes are common. There is however no clear understanding exactly to which extent, in quantitative terms, attributes are conflicting. Moreover, there is also a lack of support for handling conflicts.

3. Positive relations are primarily identified in the research literature. Industry has a clear focus on the negative influences. This is maybe not so surprising since the negative conflicts have to be addressed. However, the potential benefit of positive relations could maybe be further exploited by industry.

4. As stated in item 2, there is basically no clear picture of how much different attributes affect each other. There is a need to further address this even if it may not be possible to say an exact figure. In particular, it should be beneficial to know the approximate size of the effect. Are we talking about a factor 2 or 10?

5. Trade-offs between different attributes were obvious to the people participating in the industrial survey. It is an intrinsic part of software development and a constant balance to try to optimize management aspects (e.g. cost and timeliness in delivery), customer aspects (e.g. performance and reliability) and development aspects (e.g. maintainability and correctness).

In summary, the interviewees were almost in complete agreement that additional knowledge would be helpful, and that this knowledge is useful in order to make better decisions both at project level and at higher management level. However, the interviewees do not really find the support in the literature since the statements are too general. Thus, it can be concluded that there is a need to fill the gap between generic knowledge and the tacit knowledge in industry that is very system specific. This gap is certainly a major challenge for researchers in software quality and in particular for those addressing the trade-offs and conflicts between different software quality attributes.

References

Babbie, E. (1990) Survey Research Method, 2nd edition, Wadsworth Publishing Company, Belmont.

Boehm, B. and In, H. (1996) Identifying Quality Requirement Conflicts, IEEE Software, 13(2), 25-35.

Bosch, J. (2000) Design & Use of Software Architecture, Addison-Wesley, Kent.

Chung, L., Nixon B.A., Yu, E. and Mylopoulos, J. (2000) Non-functional Requirements in Software Engineering, Kluwer Academic Publishers, Boston.

Henningsson, K. (2001) Trade-offs and Conflicts Between Quality Attributes, Master thesis, Dept. of Software Engineering and Computer Science, Blekinge Institute of Technology, see also http://www.ipd.bth.se/khe/.

Häggander, D., Bengtsson, PO., Bosch, J. and Lundberg, L. (1999) Maintainability Myth Causes Performance Problems in Parallel Applications, in 3rd Int. IASTED Conf. on Software Engineering and Applications, Scottsdale, USA.

Kotonya, G. and Sommerville, I. (1997) Requirements Engineering, John Wiley & Sons, Devon.

McCall, J.A. (1994) Encyclopedia of Software Engineering, Vol 2, John Wiley & Sons, New York.

Svahnberg, M., and Wohlin, C. (2001) Evaluation of Software Quality Aspects for Architectural Structures using the Analytical Hierarchy Process Submitted.

Robson, C. (1993) Real World Research, Blackwell, Cornwall.