Alignment of Software Product Quality Goals in Two Outsourcing Relationships

Sebastian Barney
Blekinge Institute of Technology
Sweden
sebastian.barney@bth.se

Claes Wohlin
Blekinge Institute of Technology
Sweden
claes.wohlin@bth.se

Background: Issues with software product quality are commonly reported when organisations engage in outsourcing relationships. To address this issue, value-based software engineering literature emphasises the need for all success-critical stakeholder groups to work towards a mutually agreed goal.
Aim: This paper presents a case study that aims to compare and contrast the priority two groups place on software product quality — stakeholders internal to the development organisation, and stakeholders from outsourcing relationships.
Method: A model of software product quality was developed and used for this study based on ISO 9126 standard. Questionnaires were collected from 38 representatives of the two key stakeholder groups, in which each person rates the relative importance of aspects of software product quality using the hierarchical cumulative voting (HCV) technique. The results of these two groups were then analysed and compared.
Results: The results show the stakeholders priorities to be a merging of the priorities from both the software development organsiation, and the firm providing the outsourced services. Further, stakeholders from outsourced relationships had greater difficulty define an ideal future balance of software product qualities.
Conclusions: One of the keys to success when outsourcing is to ensure both the internal and external groups understand the needs of each other — and ensure they can work towards a sufficiently compatible goal. It may be necessary to change the way work is outsourced to align the goals of both firms to be compatible.

Quality, Outsourcing, Priorities, Case Study

1. INTRODUCTION

Many organisations are becoming more global by either establishing themselves in different countries or outsourcing to sub-contractors, for example in China (Barney, et al. 2009). The reasons for this are different, but include cost reduction, proximity to market and making use of the competence around the world (Smite, et al. 2009). This puts new requirements on software development, including different aspects on coordination, communication and control (Ågerfalk, et al. 2005). Furthermore, outsourcing may or may not include geographical, temporal and cultural challenges depending on the locations of the parties involved. While there are many benefits from outsourcing, this change has not been without problems.

One of the main challenges faced when outsourcing is ensuring a sufficient level of quality is developed in the software produced by the outsourced developers (Morgan 2004). Thus, one part is to be able to communicate the priorities on different qualities between parties involved for example from a customer to a supplier. The qualities include both product qualities such as functionality, reliability, security and maintainability as well as more project-oriented attribute such as cost and delivery date. It was hypothesised that minimizing the challenges when it comes to geographical, temporal and cultural differences would be a more favourable situation than having customer and sub-contractor far apart, i.e. it ought to be easier to communicate how to balance and prioritise different qualities.

The benefits of having of having a common understanding is highlighted in research by Phongpaibul and Boehm (2005). They find that a common understanding of software product quality is the most effective way for multiple groups to be successful in achieving a common goal. Initially the objective was to study the alignment between internal groups within a company. Thus, a method for studying the balance and priority of different company internal groups such as development, testing and different management roles was created (Barney and Wohlin 2009). It was created to determine the degree to which internal success-critical stakeholders are aligned on priorities given to various aspects of

software product quality as well as other aspects such as time and cost. However, it was realized that the method could be used also to understand the alignment between customer and sub-contractors.

Thus, this paper extends the previous work to assess the alignment between internal and outsourced development effort with regard to the priority placed on software related qualities. Subcontracted developers from two separate firms who are both working on the same product as the internal developers and co-located with the internal developers were interviewed as part of this research. This paper extends the methodology and case study previously presented to both:

- Determine the level of alignment between the internal and outsourced stakeholders, and
- Explore the reasons for the differences in software product quality priorities between these two groups.

The remainder of this paper is structured as follows. Key literature introducing the topic is provided in Section 2. Section 3 presents the research question and methodology used in this paper. Details of the case study are provided in Section 4. The results are in Section 5, with a discussion of the results in Section 6. Conclusions are presented in Section 7.

2. BACKGROUND

This section introduces key concepts and work related to this paper. First software product quality is defined in Section 2.1, with models that describe the aspects that make up software product quality presented in Section 2.2. The models of software product quality are important to this work as the alignment of the stakeholders groups is checked against a such a model. Using the broader definions of software product quality presented in the models, Section 2.3 examines some of the issues of software product quality when outsourcing development effort. Finally, recognising that there are problems of alignment between internal and external stakeholder groups, Section 2.4 presents methods for merging and harmonizing conflicting priorities in the development of software.

2.1. Software Product Quality

The most commonly used definitions used in software engineering domain define the users' perspective — 'fit for purpose' — and the manufacturing perspective — 'conformance to specification' (Hoyer and Hoyer 2001; Kitchenham and Pfleeger 1996). However, there is an increasing body of literature that recognises the value of the many perspectives involved in the development of a software product. This is presented most clearly in the value-based approach, which requires the success-critical stakeholders to come to a mutual consensus on the best way to move forward with the development of the product (Boehm and Ross 1989; Boehm and Jain 2006).

The value-based approach does not define quality as something absolute. It must be defined for a specific context or instance (Kitchenham and Pfleeger 1996). This means the optimal level of quality may not be perfect (Yourdon 1995), but how much less than perfect can only be decided in a given business context (Kitchenham and Pfleeger 1996). One of the risks with outsourcing is that the parties involved may have conflicting notions of what constitutes an acceptable level of quality for the product on which they are working together. Thus it is the value-based definition of software product quality that is used in this paper.

2.2. Models of Software Product Quality

There are a number of models of software product quality that help define and manage quality. Common models include McCall's quality model, Boehm's quality model, Dromey's quality model and ISO 9126. With the exception of Dromey's model, each of these models defines software product quality in a multi-level hierarchy.

ISO 9126 (2001) has three layers with six top-level characteristics, 20 sub-characteristics and indicators. In addition to the internal and external quality characteristics presented in Boehm and McCall's quality models, ISO 9126 also presents aspects of quality related to functionality (Milicic 2005). However, the major criticism of ISO 9126 is that it does not clearly state how the qualities it defines can be measured (Kitchenham and Pfleeger 1996).

2.3. Software Product Quality and Outsourcing

Common models and definitions of software product quality improve the likelihood of success when a company outsources software development (Phongpaibul and Boehm 2005). These things have been found more effective than definitions of quality requiring compliance to processes or specifications.

This is an important finding, as many organisations that have chosen outsourcing to reduce costs have found that decreases in quality were a common side-effect (Morgan 2004). For example, Capiluppi et al. (2006) found that the complexity of code produced by outsourced developers is higher than developers internal to the organisation developing the software.

One of the reasons a for lower software product quality in cases of outsourcing is a lack of trust between the two organisations (Moe and Smite 2007). Boehm et al. (2008) are calling for more research into the affect of cultural issues on software product quality

given the rise of outsourcing. However, the work of Phongpaibul and Boehm (2005) suggests the lack of a common understanding of software product quality alone is causing problems.

Ultimately if outsourced developers skimp on quality, then it is a loose-loose situation (Boehm and Jain 2006). The development companies do not get what they need, and the outsourced developers fail to meet the expectations of their employers.

This research to date emphasises the need for all developers working on a software product to have a common set of priorities with respect to software product quality. Hence there is a need to check the degree to which internal and external parties are aligned with the priorities they each hold.

2.4. Merging Perspectives on Software Product Quality

There are a number of methods used to help reconcile conflicting stakeholders priorities. These methods include expert judgment, the NFR Framework, Quality Functional Deployment and Theory-W.

Central to resolving conflict in the value-based approach to software engineering is Theory-W (Boehm and Ross 1989; Boehm and Jain 2006). The aim of Theory W is to create a win-win scenario for all stakeholders by:

1. Identifying the success-critical stakeholders;
2. Eliciting the requirements of these groups;
3. Creating a win-win situation by negotiating with these groups; and
4. Realising the negotiated solution through a controlled process.

The research presented in this paper seeks to help reconcile conflicts in software product quality priorities between internal and outsourced stakeholders by applying Theory-W.

3. METHODOLOGY

The methodology presented in this paper seeks to understand the priorities of internal and external stakeholders on aspects of software product quality. This section presents the research objectives, questions, and method used to conduct the research presented in this paper.

3.1. Research Objectives and Questions

There are many aspects of software product quality, and many groups that influence software product quality. While the authors' research has focused on co-located internal success-critical stakeholders to date (Barney and Wohlin 2009), companies are increasingly outsourcing software development to supplement and complement internal development effort. The change to outsourced developerment has been found to impact the quality of a software product negatively, while difficulties in communicating quality requirements are well documented (Morgan 2004).

The objective of this case study is to determine the degree to which outsourced developers are aligned with internal success-critical stakeholders in how they perceive priorities on software product quality.

It is valuable to understand both if there is a common understanding of both what the priorities *are* today, and what the priorities *should ideally be* today. Thus the research question has been broken into two sub-questions — to understand the alignment of the stakeholder groups with respect to software product quality:

- **RQ1**: As they perceive the priorities *are* today.
- **RQ2**: As they perceive the priorities *should ideally be* today.

3.2. Method

In the study presented in this paper the authors follow a methodology they previously developed and tested to determine the alignment of internal success-critical stakeholder groups with respect to software product quality (Barney and Wohlin 2009). Given the success of this method in this domain, it has been reused.

The method draws heavily on early phases Theory-W (Boehm and Ross 1989; Boehm and Jain 2006). The aim of Theory-W is to create a win-win scenario between the success-critical stakeholders, the people upon whom the success of a project is dependent. Theory-W states that in order to achieve such a scenario one must identify the success-critical stakeholders, identify how they want to win, negotiate win-win plans, and control the process to achieve the win-win situation.

They key steps in method used to answer the research question have been developed from Barney and Wohlin (2009) and are detailed in the following seven subsections.

3.2.1. Select a company and product

As each product can have different quality requirements, it is essential to ensure the study is sufficiently focused. The case study presented in this paper extends the work previously done by (Barney and Wohlin 2009), focusing on the same product.

3.2.2. Identify success-critical stakeholder groups

Success-critical stakeholder groups are groups upon whom the success of the product depends — for

example, product managers and developers. This ensures the most important perspectives are covered, while less important perspectives cannot dominate.

3.2.3. Develop quality model
The literature on software product quality recognizes that quality depends both on the perspective of the observer and the actual software product in question. As such, using any model as it appears in the literature risks not adequately defining quality in the context being studied. To use one of the quality models briefly introduced in Section 2.2 is a good starting point, but company specific needs have to be taken into account, as illustrated in the authors' previous research (Barney and Wohlin 2009).

3.2.4. Develop a questionnaire
This method proposes the use of the hierarchical cumulative voting technique (HCV) (Berander and Jönsson 2006) to elicit the relative importance of each aspect of software product quality. This method allows respondents to state the relative importance of the aspects being studied. Past research has shown that respondents have trouble comparing some aspects of software product quality at a low level directly Barney and Wohlin (2009); HCV provides a method for breaking the problem into a series of smaller direct comparison exercise, with a method to join these results back together.

3.2.5. Conduct the questionnaire
The required information should be collected by getting representatives of each of the identified success-critical stakeholder groups to complete the questionnaire. Doing this in a one-on-one structured interview allows richer information to be collected. This also helps ensure participants have a common understandings of the questionnaire with the interviewer able to assist with questions or problems faced by the respondents.

3.2.6. Analyse the results
Using the method developed by Berander and Jönsson (2006) it is possible to turn the results of the HCV exercise into cumulative voting (CV) results. From here it is possible to group the results by success-critical stakeholder group, and average the points awarded to each aspect of software product quality. These results can be used to calculate Spearman rank correlation coefficients, determining the level of alignment between the success-critical stakeholder groups.

3.2.7. Workshop the results
Presenting the results to participants from each of the success-critical stakeholder groups and asking for their response allows for a deeper understanding of the results. The following questions should be answered as best as possible:

- Do these results look reasonable?
- Do the differences make sense?
- Why do these differences exist?

4. CASE STUDY

The paper presents a case study for one of the major products at Ericsson. Ericsson is a world leading company in telecommunications, providing a wide range of products and services. These are developed and sold as generic solutions to an open market, although customized versions of the products are also developed for key customers.

An exploratory case study has been employed for this research, as it seeks to gain insights and understanding of the current situation within Ericsson (Runeson and Höst 2009).

The study was conducted in two phases, with the second phase being the focus of this paper. Each phase is described in more detail in the following subsections.

4.1. Phase 1: Internal Stakeholders

The first phase of the case study was conducted in autumn 2007 (Barney and Wohlin 2009). It was focused on the alignment of internal success-critical stakeholders with respect to software product quality. The success-critical stakeholder identified in this study were:

- *Strategic Product Managers (SPM)* have the strategic product responsibility and decides the overall product development direction.
- *Project Managers (PM)* are responsible for planning and executing projects aligned with the priorities of the strategic product management.
- *Tactical Product Managers (TPM)* supports the strategic product managers with expert knowledge of the systems and their architecture. It is also responsible for providing analysis of pre-project requirements in the form of feasibility, impact and technical dependencies.
- *Developers and Testers (R&D)* are responsible for the implementation, verification and validation of requirements.

A model of software product quality was created specifically for this study. The model was developed over a series of three two-hour meetings between one of the authors and representitives of each internal success-critical stakeholder group. The model was based on both the ISO 9126 Standard (ISO 9126 2001) and the control variables in software development identified by Beck (2000) — functionality, quality, cost and time. A number of changes were made to the ISO 9126 model to make it more meaningful and useful in the organisational setting being studied. Beck's control variables were used

to recognise the business context in which software development occurs.

A questionnaire was then developed using the model, with HCV used to determine the relative priority of each aspect of software product quality — both as perceived today and in the ideal situation. The model of software product quality and questionnaire used in the study is available online (Barney and Wohlin 2008).

In total 31 complete responses were obtained from this first phase, with between four and twelve responses per group. The results found the groups to be aligned in the priority they placed on the various aspects of software product quality both today and their perceived ideal situation (Barney and Wohlin 2009). Further the groups perceived only small changes needed to be made to the priorities in place today.

4.2. Phase 2: External Stakeholders

During the first phase all of the software development was done internally by Ericsson employees for the product studied. However, in the year following the first phase some of the development effort was outsourced to a number of consultancy firms, as shown in Figure 1, with these subcontracted providers placing self-contained teams onsite in the Ericsson office.

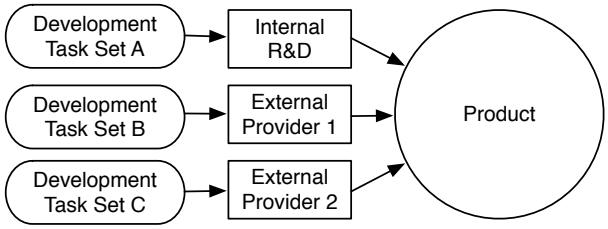

Figure 1: *Development shared between internal and external stakeholders*

Prior to this change, new developers were brought into existing teams. This meant new developers were surrounded by the support, knowledge and experience of developers with a deep understanding the product. However, the new subcontracted teams often comprised of people with no previous experience of the product. While they had experts within the organisation they could call upon for support, these people were not part of their development teams and worked with many groups.

Understanding the difficulty in communicating software product quality priorities, senior management at Ericsson wanted to confirm if the organisation was able to successfully communicate their priorities to the subcontracted teams. These subcontracted software development providers clearly had the ability to impact the software product quality, but given the difficulty in expressing quality requirements, management within Ericsson sought to determine if they were successful in achieving this aim.

To determine if the subcontractors were aligned with the internal success-critical stakeholder groups a second phase was planned to follow the first. This second phase was focused on the same product as in Phase 1. The list of success-critical stakeholder groups was expanded to include two of the subcontracted providers — referred to as *Provider 1* and *Provider 2* in this paper. A decision was made to use the model of software product quality and questionnaire from Phase 1, as both were found to work effectively and this choice allows comparisons between the two studies.

A decision was made to reuse the results of the internal success-critical stakeholders from Phase 1. A related study showed the priorities of these internal groups to be unchanged in autumn 2008 (Barney, et al. 2009).

Interviews for the second phase were conducted during spring 2009 — six months after the subcontracted developers started working on the Ericsson product studied. In total eight people were selected to participate in the study — four from each of the selected subcontracted firms. One person was unable to participate, with seven results being used in the analysis presented in this paper. The seven participants had all worked on the product at Ericsson for six months. One participant from each firm was a recent graduate, with less than 12 months industrial experience. The remaining five developers had between three and six years of experience in the software development industry.

Figure 2 shows the scope of the two studies in relation to the groups examined.

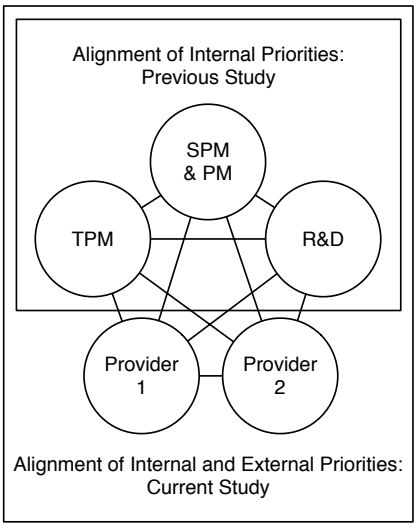

Figure 2: *Study Phases and Participant Groups*

5. RESULTS

The objective of this study is to determine the degree to which internal stakeholder groups and subcontracted

software developers are aligned in the priorities they place on aspects of software product quality. This results section brings together previous research into the alignment of internal stakeholder groups (Barney and Wohlin 2009), with new data collected on the priorities of external stakeholder groups, as shown in Figure 2. Bringing this information together allows comparisons between both sets of groups.

The research was broken into two sub-questions, presented in Section 3.1. RQ1 sought to determine the degree to which the internal and subcontracted success-critical stakeholder groups are aligned in how they perceive the priorities on aspects of software product quality *today*. The results to this sub-question are presented in Table 1. The table contains Spearman rank correlation coefficients for each possible pairs of groups. In this table the number one indicates complete agreement between two groups in their priorities on software product quality, while minus-one indicates two groups think the priorities of the other group are completely back-to-front. The results of the two subcontracted software development service providers are shown individually, and are also shown collectively in the *subcontracted* column.

The internal success-critical stakeholders are in strong agreement as to what the current priorities are on software product quality (Barney and Wohlin 2009), with correlation coefficients between 0.80 and 0.90.

The subcontracted software providers also identified a very similar set of priorities on software product quality to the internal groups in the situation as it they perceive it today, although not as closely aligned as the internal stakeholder groups. Comparing the priorities of Provider 1 against the internal stakeholder groups obtained correlation coefficients of 0.64 to 0.68, while Provider 2 obtained results between 0.65 and 0.79.

There were a number of noteworthy differences between the priorities of the internal and subcontracted stakeholder groups studied. The subcontracted groups ranked the qualities *testability*, *performance management/statistics* and *cost* higher than the internal groups. The subcontracted groups ranked *resource behaviour*, *time* and *upgradability/replaceability* lower than the internal groups. Possible reasons for these differences are discussed in Section 6 of this paper.

The second research question, RQ2, sought to understand how each success-critical stakeholder groups thought the aspects of software product quality should be prioritised today, and to what degree these groups were aligned. The results to this sub-question are presented in Table 2. These results have the same format as Table 1, with Spearman rank correlation coefficients showing the degree of alignment between the different success-critical stakeholder groups.

The internal groups were relatively well aligned (Barney and Wohlin 2009), with Spearman rank correlation coefficients between 0.65 and 0.74. The differences in the priorities of these groups can be explained — each internal group prioritised a subset of qualities higher than the other internal groups based on what most directly impacted them. For example, the R&D group prioritised maintainability higher than the other groups, as they are the only group that directly with the code.

The results showed the subcontracted software providers to be not as clearly aligned with the internal groups, as the internal groups are with each other. However, examining the priorities of the subcontracted software providers in more detail showed the member of this group to be in disagreement with each other as to what should be important.

Despite the inconsistency between the perceived ideal priorities of the outsourced software providers, there were some priorities for which this group shared a consistent vision. The outsourced software providers would like to see:

- A much greater focus on *testability*;
- A greater focus on *changeability*; and
- Less focus on *installability*.

Possible reasons for an emphasis on these aspects of software product quality are discussed in Section 6 of this paper.

6. DISCUSSION

This study sought to understand if Ericsson was able to successfully communicate its priorities with respect to software product quality to subcontracted software development providers. The results were positive for Ericsson, with the subcontracted software development providers being reasonably well aligned with the internal success-critical stakeholder groups in their priority on aspects of software product quality. However, a number of interesting points were found in the results that will be examined in this section.

6.1. Merging of Priorities

The priorities of the the subcontracted developers are a combination of the priorities of both Ericsson and the firms providing the subcontracted services, and are shaped by the way they are involved in the software development process. The results clearly indicate the subcontracted developers can identify the priorities placed on software product quality by the internal stakeholders today with correlation coefficients between 0.64 and 0.79 when comparing the priorities of the two groups. However, there were a number of consistent differences in priorities, which show the subcontracted

Table 1: *Correlation matrix showing the degree to which the groups are aligned in how they perceive the priorities today*

	SPM & PM	TPM	R&D	Subcontracted	Provider 1	Provider 2
SPM & PM	1.00	0.80	0.90	0.77	0.64	0.79
TPM		1.00	0.86	0.73	0.64	0.65
R&D			1.00	0.79	0.68	0.73
Subcontracted				1.00		
Provider 1					1.00	0.66
Provider 2						1.00

Table 2: *Correlation matrix showing the degree to which the groups are aligned in how they perceive the priorities should be today (ideal)*

	SPM & PM	TPM	R&D	Subcontracted	Provider 1	Provider 2
SPM & PM	1.00	0.74	0.71	0.03	– 0.33	0.36
TPM		1.00	0.65	0.41	– 0.11	0.62
R&D			1.00	0.37	– 0.06	0.62
Subcontracted				1.00		
Provider 1					1.00	– 0.11
Provider 2						1.00

developers adopting the priorities of the firms providing the subcontracted services.

The priorities of the firms providing the subcontracted developers are most visible where there is a consistent difference between the priorities of the internal stakeholders and the subcontracted stakeholders. The key differences in the case study presented in this paper are listed here with explanations illicited from follow-up workshops that were used to review the results.

Testability and *performance management/statistics* are two aspects of quality given a much higher priority by the subcontractors than the internal stakeholders. In order to allow Ericsson to easily see that the subcontracted teams are doing good work, these teams must make their work easy to assess. These aspects are more important to the subcontracted development teams as they have a greater need to prove themselves to secure their position than internal development teams.

Another interesting differences relates to the project management control variables of *time* and *cost*. These two variables are very closely related as the time spent on software engineering activities accounts for much of the costs of developing software. However, where internal stakeholders prioritise *time*, the subcontracted developers prioritise *cost*. The reasons for this are clear, with internal stakeholder providing and working to time-based estimates, and subcontracted developers providing and working to cost-based estimates. The subcontracted developers know the allocated budget to different activities.

Further, two aspects of software product quality were given a lower priority by the subcontracted developers than internal stakeholders — *resource behaviour* and *upgradability/replaceability*. The subcontracted developers are given the responsibility for delivering functionality, but are not involved with customer delivery. As such they do not directly face some of the issues dealt with by the internal stakeholder groups.

In summary, it seems like the subcontracted developers have a combination of the internal view of their customer and their own priorities to show their value.

6.2. The Ideal Situation

In addition to asking participants to identify the priorities today, the survey also asked participants to identify what they felt the priorities should be today. In this section of the questionnaire the internal participant groups indicated they wanted largely the same priorities as they perceived in the situation today, but with a greater emphasis on the aspects of quality that most affected their group (Barney and Wohlin 2009). For example, the developers wanted to see the aspects of maintainability given a higher priority than either their perception of the situation today or the other groups perception of the ideal situation. This shows that the participants were able to differentiate between what they perceived the priorities to be today, and what they wanted the priorities to be today.

The results of Phase 2 show that the subcontracted developers would also like some change to the current situation. Unlike the internal stakeholders, however, the subcontracted developers do not share a common vision of what priorities would be most beneficial for the product studied. Possible reasons for this result were elicited during the workshops to understand the results:

- The subcontracted developers do not have a complete vision of the product, target market or customers. They are excluded from certain aspects of product development and are not included in strategic planning. This means that they cannot approach product development from a holistic perspective. However, groups internal to the software organisation have more experience dealing with the whole product.
- The current priorities have been institutionalised in the internal stakeholders ways over an extended period of time, so the internal groups are more cautious of change.

However, the subcontracted developers were consistent on three aspects of software product quality. They would like to see a greater emphasis on *testability* and *changeability*, but less emphasis on *installability*. *Testability* is perceived undervalued by the subcontractors as they want to have accountability and prove themselves professionally to Ericsson. The need for a greater emphasis on *changeability* is justified as this would make their development tasks easier to complete. However, the subcontracted developers are not involved in the set-up of specific systems, so would not face the consequences of a reduced focus on *installability*.

These consistent differences emphasize the alternate agendas of the internal and subcontracted groups. While these differences have not caused problems in the case studied, it is possible for them to conflict and cause problems.

6.3. Validity Threats

The authors were limited in the number of responses they could collect representing the views of subcontractors. Thus the results are not necessarily representative of this group. However, all but one of the selected respondents participated in the study and the results for the situation today was described consistently between the participants.

Subcontracted developers were not included in the development of the quality model used to describe this product as this role did not exist at the time the model was developed. This makes it possible that some view on quality has been missed that relates directly to the subcontracted role. However, this study is primarily concerned with the degree to which the subcontracted stakeholders understand the priorities of software product quality of the internal stakeholders and their understanding of software product quality. Further, all results were collected in a one-on-one interview situation, with the participants given the opportunity to provide general comments and feedback.

As an exploratory case study was conducted the specific results of this study may have limited generalizability (Runeson and Höst 2009). The authors believe the priorities placed on aspects of software product quality by different groups involved in the development process will not only vary between different software products, but also throughout the life of a software product. However, the method can be applied to other cases, with more results providing the possibility for more general findings.

7. CONCLUSION

This paper aimed to understand the priorities of subcontracted software developers with respect to software product quality, and if they are aligned with the priorities of internal stakeholder groups working on the same product. The paper presents a case study of one of the major products at Ericsson. While positive, the results show a need to take great care in managing the quality output of outsourced development to achieve a desired outcome.

The results show that Ericsson has successfully integrated the subcontracted developers into the company as they have a strong understanding of the current priorities with respect to software product quality. However, it does not appear that the subcontracted developers always understand why this balance is desired. The subcontracted developers do see opportunities for improvement, but their lack of vision of the entire product, process and market means their stated preferences do not take into consideration all relevant issues.

Further, the priorities of the outsourced developers represented a merging of the priorities of the company developing the product and the firm providing the subcontracted development services. Areas affecting how the subcontracted developers would be assessed were prioritised higher by this group than the internal groups. Some aspects of quality were down-prioritised by the subcontracted developer as the developers were not involved in activities that benefited from higher quality in these areas.

The organisational setting presented in this paper, co-located task-based outsourcing, allows for a closer working relationship between the internal and external groups than other forms of outsourcing, near-shoring and off-shoring — primarily as all developers are working at the one site. In other types of outsourcing, near-shoring and off-shoring relationships it is likely to be more difficult for a company to impart their software product quality priorities. This means the partner organisation is likely to develop to their own priorities, which may not necessarily be aligned with those of the development company.

An interesting area of future research is to look at other relationships — such as outsourcing, near-shoring and

off-shoring. A greater understanding of the priorities stakeholders in these groups have in developing software will lead to more effective relationships and software development.

The research by Phongpaibul and Boehm (2005) found that a common understanding of software product quality priorities between groups was a more effective way of achieving the desired level of quality than through process or product descriptions. Thus it is important for an organisation to ensure these internal and outsourced workers share a common value system with respect to software product quality. The method presented in this paper has been demonstrated to be able to elicit and compare the priorities of numerous groups. The authors' experience has shown this is a necessary step in getting the groups to understand each other better, and work together effectively.

ACKNOWLEDGMENTS

We would like to thank Ericsson for their active involvement in and support of this research.

This work was partly funded by the Industrial Excellence Center EASE — Embedded Applications Software Engineering, (http://ease.cs.lth.se).

8. REFERENCES

Ågerfalk, P. J., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B. and Conchúir, E. Ó. (2005) A Framework for Considering Opportunities and Threats in Distributed Software. In *Proceedings of the International Workshop on Distributed Software Development (DiSD 2005)*, Paris, 29 August, pages 47-61, Austrian Computer Society.

Barney, S., Hu, G., Aurum, A. and Wohlin, C. (2009) Creating Software Product Value in China. *IEEE Software*, 26(4):84–90.

Barney, S. and Wohlin, C. (2008) *Software product quality questionnaire*. http://www.bth.se/tek/aps/sba

Barney, S. and Wohlin, C. (2009) Software product quality: Ensuring a common goal. In Qing Wang, Ray Madachy, and Dietmar Pfahl, editors, *Proceedings of the International Conference on Software Process (ICSP)*, pages 256–267, May.

Barney, S., Wohlin, C. and Aurum, A. (2009) Balancing software product investments. In *Empirical Software Engineering and Management (ESEM)*, pages 257–268, October.

Beck, K. (2000) Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading, Massachusetts.

Berander, P. and Jönsson, P. (2006) Hierarchical cumulative voting (HCV) — Prioritization of requirements in hierarchies. *International Journal of Software Engineering and Knowledge Engineering*, 16(6):819–849, December.

Boehm, B., Chulani, S., Verner, J. and Wong, B. (2008) Sixth workshop on software quality. In *ICSE Companion '08: Companion of the 30th international conference on Software engineering*, pages 1035–1036, New York, NY, USA. ACM.

Boehm, B. and Jain, A. (2006) An initial theory of value-based software engineering. *Value-Based Software Engineering*, pages 15–37.

Boehm, B. W. and Ross, R. (1989) Theory-w software project management principles and examples. *IEEE Transactions on Software Engineering*, 15(7):902–916.

Capiluppi, A. Millen, J. and Boldyreff, C. (2006) How outsourcing affects the quality of mission critical software. In *13th Working Conference on Reverse Engineering (WCRE)*, pages 285–287, October.

Hoyer, R. W. and Hoyer, B. B. Y. (2001) What is quality? *Quality Progress*, 34(7):53–62, July.

ISO9126 (2001) *Software Engineering — Product Quality — Part 1: Quality Model*. International Standards Organization.

Kitchenham, B. and Pfleeger, S. L. (1996) Software quality: The elusive target. *IEEE Software*, 13(1):12–21, January.

Milicic, D. (2005) Software Quality Attributes and Trade-Offs, chapter *Software Quality Models and Philosophies*, pages 3–19. Blekinge Institute of Technology.

Moe, N. and Smite, D. (2007) Understanding lacking trust in global software teams: A multi-case study. *Product-Focused Software Process Improvement*, pages 20–34.

Morgan, L. (2004) Consider the outsource. *Software Development Times*, (100):28–30, Apr.

Phongpaibul, M. and Boehm, B. (2005) Improving quality through software process improvement in Thailand: initial analysis. In *3-WoSQ: Proceedings of the third workshop on Software quality*, pages 1–6, New York, NY, USA. ACM.

Runeson, P. and Höst, M. (2009) Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, April 2009.

Smite, D., Wohlin, C., Gorschek, T. and Feldt, R. (2009) Empirical Evidence in Global Software Engineering: A Systematic Review. Accepted for publication in *Empirical Software Engineering: An International Journal*. DOI: 10.1007/s10664-009-9123-y (available in Online First).

Yourdon, E. (1995) When good enough software is best. *IEEE Software*, 12(3):79–81, May 1995.