Methods for Supporting Management of Interactions Between Quality Characteristics

Gabriel Alberto García-Mireles[1], Mª Ángeles Moraga de la Rubia[2], Félix García[2] and Mario Piattini[2]
[1]Departamento de Matemáticas, Universidad de Sonora, Blvrd. Rosales y Rodríguez s/n, Hermosillo, Sonora, Mexico
[2]Instituto de Tecnologías y Sistemas de Información, Universidad de Castilla-La Mancha, Ciudad Real, Spain

Keywords: Software Quality Characteristics, Interaction between Quality Characteristics, Mapping Study.

Abstract: Improving a particular quality characteristic in a software product may have a negative impact on the others. The literature shows that few organizations handle interactions between quality characteristics; this neglect may be a causal factor in failed projects. That led us to propose a process framework to support organizations that want to manage the interactions between quality characteristics. In this paper, we present the methods that may be used when the process framework is deployed. The methods were extracted from a published mapping study on software quality trade-offs. They were classified with regard to the particular context facet addressed and the specific decision-making approach used. Our contribution is a set of methods to manage interaction between quality characteristics, organized into a software process framework.

1 INTRODUCTION

When developing software, the enhancement of a given quality characteristic may have a negative impact on the others (Ashrafi, 2003). The advantages and disadvantages of each solution option should be analyzed in order to minimize negative collateral effects. The lack of management of interactions between quality characteristics can be a causal factor in failed software projects (Thakurta, 2012); (Theofanos and Pfleeger, 2011).

Software organizations need adequate processes to manage this kind of interactions. It is nonetheless the case that product quality, as defined by standards such as the ISO9126, is barely addressed in software process improvement literature (Unterkalmsteiner et al., 2012). In a previous paper, and in an effort to support software organizations that want to deal with interaction between software quality characteristics, we proposed a process framework to manage interactions between quality characteristics (García-Mireles et al., 2013b).

Our goal in this paper is thus to identify an initial set of methods that software organizations may implement to manage the interactions between quality requirements. The objective is to answer the following research questions:

RQ1. What methods can a software organization use to manage interactions between quality characteristics?

RQ2. What particular quality models are considered in these methods?

The research relies both on reviewing 20 empirical papers of a mapping study (Barney et al., 2012) and on applying techniques to carry out mapping studies (Kitchenham and Charters, 2007). Our next step is to map methods/practices reviewed with our framework. The contribution is a set of methods or practices directed at practitioners who wish to apply a systematic approach to dealing with quality characteristic interactions.

This paper is organized as follows: Section 2 describes the work related to managing interaction at earlier stages of the software lifecycle and gives an overview of our process framework. Section 3 gives a description of the method used to study the empirical papers. Section 4 provides the categorization of articles studied in our attempt to answer the research questions. Section 5 establishes a relationship between the methods and our proposal; it also outlines some results of the survey carried out. Section 6 presents the discussion about the results. Finally, Section 7 sets out our conclusions and discusses future work.

García-Mireles G., Moraga de la Rubia M., García F. and Piattini M..
Methods for Supporting Management of Interactions Between Quality Characteristics.
DOI: 10.5220/0004867400930100
In *Proceedings of the 9th International Conference on Evaluation of Novel Approaches to Software Engineering* (ENASE-2014), pages 93-100
ISBN: 978-989-758-030-7

2 RELATED WORK

Interaction between requirements may arise in situations in which one requirement places constraints on the design or coding options (Dahlstedt and Persson, 2005). To select a solution option, there are prioritization and negotiation approaches (Lehtola and Kauppinen, 2004).

On the one hand, prioritization approaches assign weight to each relevant criterion when assessing requirements/solution options (Berander and Andrews, 2005). On the other hand, in a negotiation approach the stakeholders look for an agreement in which the conflict between goals is resolved (Grünbacher and Seyff, 2005).

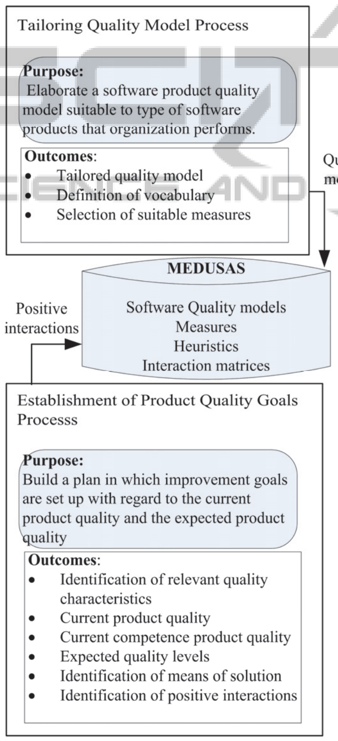

Figure 1: Partial view of the process framework.

Decisions about quality characteristics that must be considered in a software project have an impact on software process (Allen et al., 2006). Software process literature, however, pays little attention to product quality characteristics (Unterkalmsteiner et al., 2012). Indeed, Chiam et al., (2013) argued that there are no systematic approaches to represent and integrate methods that support product quality attributes within the current software process models. Bearing all this in mind, in a previous work we presented a process framework for managing interactions between quality characteristics (García-Mireles et al., 2013b).

The main goal of our process framework is to manage interactions between quality characteristics which arise during software development (García-Mireles et al., 2013b). The framework relies on a repository of tailored product quality models focused on usability, maintainability, and security. It also contains interaction matrices which describe the type of relationships that exist between quality characteristics. The initial content of these matrices is based on a review of interactions between quality characteristics (García-Mireles et al., 2013a).

The process framework is composed of several processes, which can be implemented both at organizational level and at project level. Fig. 1 shows the processes that may be applied at the organizational level to tailor quality models and to establish quality goals. A software organization which wishes to manage quality characteristic interactions should define an appropriate quality model. If clients and users are expecting an enhanced product quality, then the process to define quality goals could be performed. Each process, however, requires appropriate methods.

3 METHOD

The identification of an initial set of methods to manage the interactions between software quality characteristics was based on a review of the 20 empirical papers classified as process and requirements in the mapping study (Barney et al., 2012). This mapping study meets quality criteria (Table 1) for follow-on research activities (Kitchenham et al., 2011). After the mapping study evaluation and in order to answer our research questions (Section 1), we define a classification schema to extract data.

Table 1: Quality of the mapping study.

Criterion	Mapping study (Barney et al. 2012)
References	List of 43 empirical papers
Reliable Classification	Papers are categorized by artefact focus, rigour and relevance (Ivarsson and Gorschek, 2011)
Stringent search process	Five databases consulted Search period limited to 2005-2010 Research protocol was built Keywords identified iteratively At least two researchers reviewed papers

Our first classification is concerned with research type (Table 2). If a paper describes a technology for changing the software process, it is considered as *intervention research (I)*. Otherwise, the empirical paper belongs to the *descriptive research (D)* category.

Our second classification used the context facet categories checklist (Petersen and Wohlin, 2009). *Process* describes the work-flow of the development. *Practices, tools* and *techniques* describe systematic approaches that are used in the organization; we added the component of *method* to this category. *Roles* belong to the *people* facet. *Product* is the software system developed. In the *organization* facet, in contrast to the original checklist, we consider those studies describing issues/practices at an organizational level.

Since the *rigour* classification gives an approximate overview of the state of research (Ivarsson and Gorschek, 2011), we convert it to a binary scale. The paper has high rigour if its value is greater or equal to 2; otherwise, the rigour is considered as low.

Other columns (Table 2), such as *research method, industrial sector* and *software type* help us to understand the context in which the research was conducted. The *QM/QC* column (Table 2) is related to the second question; it contains either the quality models used or the quality characteristics mentioned.

In addition, we classify the papers with regard to the approach used in making decisions (Berander and Andrews, 2005); (Lehtola and Kauppinen, 2004). With the empirical studies now classified, we explore the relationships between the methods identified and the process framework (Fig. 1). Finally, we present some results from an exploratory survey.

4 ANALYSIS OF PUBLICATIONS

In this section we present the main findings from the empirical papers reviewed. The papers belonging to descriptive research also correspond to the organization facet category (Berntsson Svensson et al., 2009); (Barney and Wohlin, 2009); (Barney et al., 2009). The principal findings reported are that the priorities of product managers concerning quality requirements (QR) are different from those of project leaders.

To answer the research questions, we present the findings from the papers reviewed, considering the particular context facet used to classify the paper.

4.1 Tools

We found five papers whose main topic is the description of tools. In their effort to assign a priority to quality characteristics, Xiaojing and Jihong (2010) and Chang et al., (2008) based their tool on AHP and Fuzzy Logic. Fuzzy methods may reduce ambiguity and uncertainty of values assigned to software quality attributes. Assessment criteria are based on quality models. However, these authors do not discuss the impact of tools in software process.

From the negotiation approach perspective, the studies proposed by Linhares et al., (2009) and Ramires et al., (2005), introduce an argumentation-based model to support either negotiation among stakeholders or technical reviews. Patankar and Hewett (2008), for their part, present an algorithm for negotiating web services.

4.2 Methods

Six publications deal with methods. Four of them correspond to the negotiation category (Svensson et al., 2008); (Svensson et al., 2010); (Vanhanen et al., 2009); (Regnell et al., 2007). The other two belong to the prioritization approach (Lacerda et al., 2010); (Yahaya and Deraman, 2010).

Regnell et al., (2007) present the Quality Performance (QUPER) model. The method aims to support prioritization of quality aspects at early stages of release planning. It considers that a change in quality level could result in a non-linear value in either costs or benefits. The authors propose a quality view in which they identify breakpoints and barriers. Breakpoints are related to quality levels that have an impact on benefits, while barriers are related to quality levels and their respective costs. Breakpoint and cost barriers are difficult to identify when a new technology arises. Findings from validating QUPER method show that it is difficult to identify quality indicators and their respective values (Svensson et al., 2008). They also tell us that expertise in the area, the latest test outcomes and years of domain knowledge are factors which might contribute to establishing appropriate breakpoints (Svensson et al., 2010).

Vanhanen et al., (2009) propose a method to handle quality goals that is based on the Quality Attribute Workshop (QAW) and Quality Performance method (QUPER). Through a brainstorming session the quality goals are elicited and the most important quality goals are elaborated.

Lacerda et al., (2010) propose a method based on measures and a balance-scoreboard to identify business objectives for a new product portfolio, considering contextual factors. In addition, Yahaya and Deraman (2010) describe a method for assessing software which is in the operation stage. Criteria weights (based on ISO9126) for quality assessment were determined previously, by means of a survey. The method requires collaborative discussion on the part of the user, developer, and independent assessor.

4.3 Process

Of the papers which describe processes, three use a prioritization approach (Onut and Efendigil, 2010); (Trienekens et al., 2010); (Sibisi and Van Waveren, 2007). Another uses a risk-based approach (Mead and Stehney, 2005).

Onut and Efendigil (2010) propose a decision process for choosing an Enterprise Resource Planning (ERP) supplier. The decision-makers establish the priority of decision criteria using a fuzzy AHP method. Trienekens et al., (2010), on the other hand, based their proposal on two changes: redefinition of quality characteristics from a product model and using a prioritization method (AHP).

Mead and Stehney (2005) describe the experience of applying a methodology for eliciting and prioritizing security requirements in a company which manages system assets. The process considers how security requirements can be related to business goals. Based on ISO 9126, the Goal-Question-Metric (GQM) and ISO 14598, Sibisi and Van Waveren (2007) present a process framework for customizing software quality models. They develop a survey questionnaire based on measures suggested for each quality sub-characteristic.

Table 2: Empirical publications reviewed (legend: QM/QC: Quality model(s)/Quality characteristic(s). Type: I: intervention, D: descriptive).

Id	Type	Facet	QM/QC	R. Method	Industrial S.	Software Type	Rigour
(Svensson et al., 2008)	I	Method	Performance	Action Research	Telecomm.	Mobile Products	High
(Svensson et al., 2010)	I	Method	Maintainability, efficiency	Action Research	Electronic Payment	Payment Terminals	High
(Vanhanen et al., 2009)	I	Method	ISO9126	Case Study	Market Driven	N/A	High
(Regnell et al., 2007)	I	Method	N/A	Interview	Telecomm.	Mobile Product	Low
(Berntsson Svensson et al., 2009)	D	Organi-zation	ISO9126 & McCall	Interview	Telecomm & Control	Embedded Systems	High
(Barney and Wohlin, 2009)	D	Organi-zation	ISO9126 + time, cost, scope	Survey	Telecomm.	2 Products	Low
(Lacerda et al., 2010)	I	Method	N/A	Case Study	N/A	SOA-Based	Low
(Onut and Efendigil, 2010)	I	Process	ISO9126 + cost + reputation	Case Study	Manufacturing/ Chemical Industry	ERP System	High
(Barney et al., 2009)	D	Organi-zation	Features, time, cost, (ISO9126)	Survey	Telecomm.	2 Products	Low
(Mead and Stehney, 2005)	I	Process	Security	Case Study	Management Services	IT Asset Mngmt	Low
(Sibisi and Van Waveren, 2007)	I	Process	ISO9126, ISO 14598 + GQM	Survey	Entertainment	Embedded System	High
(Trienekens et al., 2010)	I	Process	ISO9126	Case Study	Naval	Mission-Critical	High
(Oliveira et al., 2008)	I	Product	Internal quality	Experiment	Control Systems	Embedded Systems	High
(Yahaya and Deraman, 2010)	I	Method	ISO9126 + integrity	Case Study	Health Sector	Information System	Low
(Fogelström et al., 2009)	D	Roles	N/A	Experiment	N/A	N/A	High
(Linhares et al., 2009)	I	Tools	N/A	Experiment	Telecomm.	N/A	Low
(Ramires et al., 2005)	I	Tools	ISO9126	Experiment	Government	Pension Systems	Low
(Xiaojing and Jihong, 2010)	I	Tools	McCall	Case Study	N/A	N/A	Low
(Chang et al., 2008)	I	Tools	ISO9126	Case Study	Government	Video Recorder System	Low
(Patankar and Hewett, 2008)	I	Tools	QoS metrics	Example	N/A	Web Services	Low

Table 3: Methods that might be used in the process framework to manage interactions between quality characteristics.

Make-decision approach	Method / technique	Purpose	Requirements / Notes
Prioritization methods	AHP Fuzzy AHP	To establish the weight of quality characteristics in evaluation criteria To assess software quality To select the best product alternative	Require tool support to capture data and to validate them The technique is time consuming Define the term which is used to prioritize
	Measures	To assess software alternatives with regard to measures taken and presented on a scoreboard or as performance indicators	Build a specific quality model. Identify suitable measures and procedures to perform measurement and data aggregation
	Surveys	To elicit software quality with regard to perception of users and developers	Build a specific quality model. Identify suitable measures and procedures to perform data aggregation Construct a questionnaire
Negotiation approach	Risk-based	To identify relevant quality requirements based on the risks related to the software product	Determine business goals and product goals Use specific methods to elicit user requirements Apply specific methods for modeling quality requirements and assessing the related risks Align product quality requirements with business goals
	Tool-based	To support argumentation among stakeholders To support negotiation in a distributed environment	A tool must be provided The tool must have an argumentation model or component to support participants' comments The tool must implement a model to support decision-making
	Workshop	To elicit quality requirements , quality indicators, quality measures, and quality values To customize quality models To compare quality of their own product with regard to market /competence	A moderator is needed to support the method Stakeholder experience in the software domain and industry sector is required to identify relevant quality indicators and quality values Standards and suppliers can help to identify quality indicators and quality values Establish a voting/ranking approach

4.4 How Quality Models Are Used

We found a broad range of reference to the term "quality" in papers reviewed. ISO9126 is cited in ten papers, while the McCall model is referred to in two publications. Some particular quality characteristics are also addressed. In addition, there are some proposals that trade-off software quality against other criteria such as time, cost, scope, intellectual capital, the supplier's reputation, or the impact factor. Some papers, however, do not mention the product quality model used as reference (Svensson et al., 2010); (Regnell et al., 2007); (Svensson et al., 2008).

Other papers addressed a measure-based approach for dealing with software quality. Lacerda et al., (2010) use quality terms without definition, but their proposal includes measures to control quality terms. Some authors used internal quality measures to assess software internal quality (Oliveira et al., 2008), or used measures to assess the quality of web service (Patankar and Hewett, 2008).

Quality models are also used, as they are defined, for classifying requirements (Ramires et al., 2005), for establishing weight for evaluation criteria in AHP (Xiaojing and Jihong, 2010); (Chang et al., 2008); (Onut and Efendigil, 2010) as a checklist to identify new quality requirements (Vanhanen et al., 2009), or as a checklist for interviewing practitioners (Berntsson Svensson et al., 2009).

Table 4: Main findings of the survey.

Aspect	Findings
Quality models	Usability and maintainability models based on ISO25010
Stakeholders	Analyst, project leader, customer representatives
Approach to trade-offs	Negotiation and code measures aggregated on a scoreboard

Other methods have adapted the ISO 9126 with regard to the specific project context, establishing specific quality indicators and specific measures (Trienekens et al., 2010). Various other researchers related quality characteristics to their respective measures, producing a questionnaire that would assess software quality (Yahaya and Deraman, 2010). In addition, one proposal set out a method for building specific product quality models, taking into account measures for assessing ISO9126 quality characteristics; it uses GQM to clarify the meaning of measures (Sibisi and Van Waveren, 2007).

5 METHODS SUPPORTING OUR PROCESS FRAMEWORK

We selected all the empirical papers categorized as intervention research because they validated the method in an organizational setting. They are presented as a summary of findings (Table 3) to classify the evidence with regard to the approach used to make decisions. The data shown in Table 3 can support the definition of a repository of methods to support the process framework for managing interactions between quality characteristics (García-Mireles et al., 2013b).

A hypothetical example can show the relationship between methods identified and the process framework. For instance, if a software organization wants to assess quality characteristics when buying a software product, it may implement the tailoring quality model process (Fig. 1). In this context it may be appropriate to consider quality models, taking into account relevant quality characteristics and surveying stakeholders to establish a relative order among the options (survey method from Table 3). In contrast, when trade-offs are required at earlier stages of software development, it is necessary to customize the quality model, identify relevant measures and quality indicators, and establish target values (measures method from Table 3).

We surveyed three small companies to understand how they deal with interactions between quality characteristics (Table 4). All of the firms based their decisions on a measure-based scoreboard which displayed values for usability and maintainability. That means that if some discrepancy about quality goals came up in the meeting, participants reviewed the data and established actions to carry out until the next meeting. The survey results, then, allow us to undertake deeper research on this topic.

6 DISCUSSION

As an answer for the first research question, regarding methods for prioritization and negotiation approaches, we found that AHP, fuzzy AHP, surveys, and measure-based methods are suggested for the prioritization of quality characteristics. Risk-based analysis, argumentation-based tools and workshops are the approaches that pertain to the negotiation approach. The identification of the purpose of methods and its requirements can support the selection of methods, considering the particular context in which software development is carried out. The mapping of these methods to our process framework organizes them with regard to organizational and project goals (García-Mireles et al., 2013b).

The method's purpose allows us to find out which process, as described in ISO/IEC 12207 (ISO, 2008), may use it. The following processes may have a relationship with the methods reviewed: acquisition process, measurement process, software review process, software operation process, stakeholder requirements definition process, system requirements analysis process, and software requirements analysis process. Since the management of interactions between quality characteristics can be considered throughout the software life cycle, our proposed framework provides a systematic approach to deal with this kind of interactions (García-Mireles et al., 2013b).

On the other hand, research question two focused on the quality models addressed. In a nutshell, quality models such as ISO9126 can be used without any change and may be employed either to evaluate perceptions about quality or to assess the general product quality. During software development, however, the quality model must be customized, based on suitable indicators and appropriate measures. The adaptation can include adding new quality sub-characteristics, removing unnecessary components or redefining quality terms. It should be said, nevertheless, that we did also find some papers which do not include the definition of quality terms; this may be an issue when the method is compared with other studies.

Limitations on the research method include the selection of primary papers and data aggregation. Despite the fact that the research period is limited to 2005-2010 (Barney et al., 2012), the mapping study offers the possibility of exploring the whole range of options for dealing with interactions between quality characteristics. This fulfils our goal of identifying an initial set of methods to manage the interaction between quality characteristics. With regard to the classification schema used, we use published classification schema that can facilitate the review of empirical papers from both a process and a decision-making perspective. Indeed, data synthesis based on classification schema has also been reported in literature reviews (Genero et al., 2011).

7 CONCLUSIONS

The software engineering community is aware of interdependencies among software quality characteristics. When conflictive interactions arise, software engineers should manage them. In order to understand how the methods to manage interactions can be used from a software process perspective, we reviewed empirical publications reported in a mapping study.

In order to answer our first question, about methods to manage interactions between quality characteristics, the literature suggests that AHP, measures, surveys, workshops and tools can be used to prioritize or negotiate quality characteristics/requirements. The goals of a particular project, as well as the resources available are relevant factors in choosing appropriate methods. In addition, the classification schema applied help in the identification of methods to support our process framework for managing interactions between quality characteristics.

In order to answer the second question, about the quality models used, we found that quality models, such as ISO9126, can be used either just as they had been defined, or customized. The particular contextual factors of the software systems and the goals to evaluate software product quality should be considered when the quality model is used or adapted.

As future work, we need to enhance the proposed framework, by considering methods and techniques that could be used at different software lifecycle stages. In this exploratory study we considered some of the methods proposed, but we will also have to review specific methods related to usability, security, and maintainability, as well as those concerning how to deal with their interdependencies.

ACKNOWLEDGEMENTS

This work has been funded by the GEODAS-BC project (Ministerio de Economía y Competitividad and Fondo Europeo de Desarrollo Regional FEDER, TIN2012-37493-C03-01).

REFERENCES

Allen, J., Kitchenham, B. & Konrad, M. 2006. Theme Q. The relationships between processes and product qualities. In: E. Forrester (eds.) A Process Research Framework. *In:* Forrester, E. (ed.) *A Process Research Framework.* Software Engineering Institute, Carnegie Mellon.19-28.

Ashrafi, N. 2003. The impact of software process improvement on quality: in theory and practice. *Information & Management* 40**,** 677-690.

Barney, S., Petersen, K., Svahnberg, M., Aurum, A. & Barney, H. 2012. Software quality trade-offs: A systematic map. *Information and Software Technology,* 54**,** 651-662.

Barney, S. & Wohlin, C. 2009. Software Product Quality: Ensuring a Common Goal. *In:* Wang, Q., Garousi, V., Madachy, R. & PFAHL, D., eds. Trustworthy Software Development Processes. Springer Berlin Heidelberg, 256-267.

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

Berander, P. & Andrews, A. 2005. Requirements Prioritization. *In:* AURUM, A. & WOHLIN, C., eds. Engineering and Managing Software Requirements. Springer Berlin Heidelberg, 69-94.

Berntsson Svensson, R., Gorschek, T. & Regnell, B. 2009. Quality requirements in practice: An interview study in requirements engineering for embedded systems. *In:* Requirements Engineering: Foundation for Software Quality. 218-232.

Chang, C. W., Wu, C. R. & Lin, H. L. 2008. Integrating fuzzy theory and hierarchy concepts to evaluate software quality. *Software Quality Journal,* 16**,** 263-276.

Chiam, Y. K., Staples, M., Ye, X. & Zhu, L. 2013. Applying a selection method to choose Quality Attribute Techniques. *Information and Software Technology,* doi: http://dx.doi.org/10.1016/j.infsof.2013.02.001.

Dahlstedt, Å. & Persson, A. 2005. Requirements Interdependencies: State of the Art and Future Challenges. *In:* AURUM, A. & WOHLIN, C., eds. Engineering and Managing Software Requirements. Springer Berlin Heidelberg, 95-116.

Fogelström, N. D., Barney, S., Aurum, A. & Hederstierna, A. 2009. When product managers gamble with requirements: Attitudes to value and risk. *In:* 15th International Working Conference on Requirements Engineering: Foundation for Software Quality (RefsQ). 1-15.

García-Mireles, G. A., Moraga, M. Á., Garcia, F. & Piattini, M. 2013a. Identificación de interacciones entre las características de calidad del software. *In:* JISBD2013, Universidad Complutense de Madrid, Madrid, España. 141-154.

García-Mireles, G. A., Moraga, M. Á., García, F. & Piattini, M. 2013b. A framework to support quality trade-offs from a process-based perspective. *In:* MCCAFFERY, F., O'CONNOR, R. V. & MESSNARZ, R., eds. EuroSPI2013,CCIS 364. Springer-Verlag Berling Heidelberg, 96-107.

Genero, M., Fernández-Saez, A. M., Nelson, H. J., Poels, G. & Piattini, M. 2011. A systematic literature review on the quality of UML models. *Journal of Database Management,* 22**,** 46-70.

Grünbacher, P. & Seyff, N. 2005. Requirements Negotiation. *In:* AURUM, A. & WOHLIN, C., eds. Engineering and Managing Software Requirements. Springer Berlin Heidelberg, 143-162.

Iso 2008. ISO/IEC 12207 Systems and software engineering — Software life cycle processes.

Ivarsson, M. & Gorschek, T. 2011. A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering,* 16**,** 365-395.

Kitchenham, B. & Charters, S. 2007. Guidelines for Performing Systematic Literature Review in Software Engineering. *ver. 2.3, Keele University, EBSE Technical Report, EBSE-2007-01*.

Kitchenham, B. A., Budgen, D. & Pearl Brereton, O. 2011. Using mapping studies as the basis for further research - A participant-observer case study. *Information and Software Technology,* 53**,** 638-651.

Lacerda, R. T. O., Ensslin, L. & Ensslin, S. R. 2010. A study case about a software project management success metrics. *In:* Software Engineering Workshop (SEW), 2009 33rd Annual IEEE 45-54.

Lehtola, L. & Kauppinen, M. 2004. Empirical evaluation of two requirements prioritization methods in product development projects. *In:* LNCS 3281. 161-170.

Linhares, G. R., Borges, M. S. & Antunes, P. 2009. Negotiation-Collaboration in Formal Technical Reviews. *In:* CARRIÇO, L., BALOIAN, N. & FONSECA, B., eds. Groupware: Design, Implementation, and Use. Springer Berlin Heidelberg, 344-356.

Mead, N. R. & Stehney, T. 2005. Security quality requirements engineering (SQUARE) methodology. *In:* Proceedings of the 2005 workshop on Software engineering for secure systems\—building trustworthy applications, St. Louis, Missouri. 1083214: ACM, 1-7.

Oliveira, M. F. S., Redin, R. M., Carro, L., Lamb, L. D. C. & Wagner, F. R. 2008. Software quality metrics and their impact on embedded software. *In:* Model-based Methodologies for Pervasive and Embedded Software, 2008. MOMPES 2008. 5th International Workshop on 68-77.

Onut, S. & Efendigil, T. 2010. A theorical model design for ERP software selection process under the constraints of cost and quality: A fuzzy approach. *Journal of Intelligent and Fuzzy Systems,* 21**,** 365-378.

Patankar, V. & Hewett, R. 2008. Automated negotiations in web service procurement. *In:* Internet and Web Applications and Services, 2008. ICIW '08. Third International Conference on 620-625.

Petersen, K. & Wohlin, C. 2009. Context in industrial software engineering research. *In:* Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on. 401-404.

Ramires, J., Antunes, P. & Respício, A. 2005. Software requirements negotiation using the software quality function deployment. *In:* Groupware: Design, Implementation, and Use. 308-324.

Regnell, B., Höst, M. & Berntsson Svensson, R. 2007. A Quality Performance Model for Cost-Benefit Analysis of Non-functional Requirements Applied to the Mobile Handset Domain. *In:* SAWYER, P., PAECH, B. & HEYMANS, P., eds. Requirements Engineering: Foundation for Software Quality. Springer Berlin Heidelberg, 277-291.

Sibisi, M. & Van Waveren, C. C. 2007. A process framework for customising software quality models. *In:* AFRICON 2007. 1-8.

Svensson, R. B., Olsson, T. & Regnell, B. 2008. Introducing Support for Release Planning of Quality Requirements- An Industrial Evaluation of the QUPER Model. *In:* Software Product Management, 2008. IWSPM '08. Second International Workshop on 18-26.

Svensson, R. B., Sprockel, Y., Regnell, B. & Brinkkemper, S. 2010. Cost and benefit analysis of quality requirements in competitive software product management: A case study on the QUPER model. *In:* Software Product Management (IWSPM), 2010 Fourth International Workshop on. 40-48.

Thakurta, R. 2012. A framework for prioritization of quality requirements for inclusion in a software project. *Software Quality Journal,* doi: 10.1007/s11219-012-9188-5**,** 1-25.

Theofanos, M. F. & Pfleeger, S. L. 2011. Guest Editors' introduction: Shouldn't all security be usable? *IEEE Security and Privacy,* 9**,** 12-17.

Trienekens, J. J. M., Kusters, R. J. & Brussel, D. C. 2010. Quality specification and metrication, results from a case-study in a mission-critical software domain. *Software Quality Journal,* 18**,** 469-490.

Unterkalmsteiner, M., Gorschek, T., Islam, A. K. M. M., Cheng, C. K., Permadi, R. B. & Feldt, R. 2012. Evaluation and Measurement of Software Process Improvement— A Systematic Literature Review. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,* 38**,** 398-424.

Vanhanen, J., Mäntylä, M. V. & Itkonen, J. 2009. Lightweight elicitation and analysis of software product quality goals - A multiple industrial case study. *In:* Software Product Management (IWSPM), 2009 Third International Workshop on. IEEE Computer Society, 42-52.

Xiaojing, L. & Jihong, P. 2010. A Fuzzy Synthetic Evaluation Method for Software Quality. *In:* e-Business and Information System Security (EBISS), 2010 2nd International Conference on. 1-4.

Yahaya, J. H. & Deraman, A. 2010. Measuring unmeasurable attributes of software quality using pragmatic quality factor. *In:* Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on. 197-202.