

# A Process Support with Which to Identify Interactions Between Quality Characteristics

Gabriel Alberto García-Mireles<sup>1(✉)</sup>, M<sup>a</sup> Ángeles Moraga<sup>2</sup>,  
Félix García<sup>2</sup>, and Mario Piattini<sup>2</sup>

<sup>1</sup> Departamento de Matemáticas,  
Universidad de Sonora, Hermosillo, Sonora, Mexico  
mireles@mat.uson.mx

<sup>2</sup> Instituto de Tecnologías y Sistemas de Información,  
Universidad de Castilla-La Mancha, Ciudad Real, Spain  
{MariaAngeles.Moraga,Felix.Garcia,  
Mario.Piattini}@uclm.es

**Abstract.** Achieving a balance between the quality characteristics that need to be addressed during the development of a software product may determine the success of a software project. However, few software organizations deal with interactions between the quality characteristics that could be present in a software project. In order to support organizations, we have developed a process framework, SQIMF, which can be used to manage this type of interactions. In this work we describe one of the SQIMF processes - that which is employed to monitor product quality requirements - in order to support software organizations as regards identifying interactions between quality requirements, in addition to characterizing them and identifying relevant contextual factors. An exploratory case study was conducted in order to initiate the validation of the proposed process, as the result of which we found interactions between usability and security during the inception phase of a software project.

**Keywords:** Software product quality · Interaction between quality characteristics · Process for monitoring quality characteristics interactions · Interaction between quality requirements · SQIMF framework

## 1 Introduction

One of the main goals of software engineering is to deliver high-quality software products and systems. The identification and specification of both functional and non-functional requirements (NFRs) are important activities as regards establishing a baseline that can be used to assess software quality. Quality requirements, as a subset of the NFRs [1], are closely related to both making decisions about selecting implementation technologies and driving the process used to design software architecture [2]. However, dealing with quality requirements during software development is difficult because they are hard to define, they are described vaguely, and they might influence each other [3]. Software developers might therefore select a design option that compromises some requirements in order to achieve others [3].

The interaction between requirements is described as a situation in which the satisfaction of one requirement may influence the satisfaction of another [4]. In this work, we are interested in the interactions between quality requirements, particularly the negative interactions, or the conflict between quality requirements. For instance, negative interactions between usability and security requirements could occur when the implementation of a means to provide access to a software system requires the users to memorize long strings of illegible data.

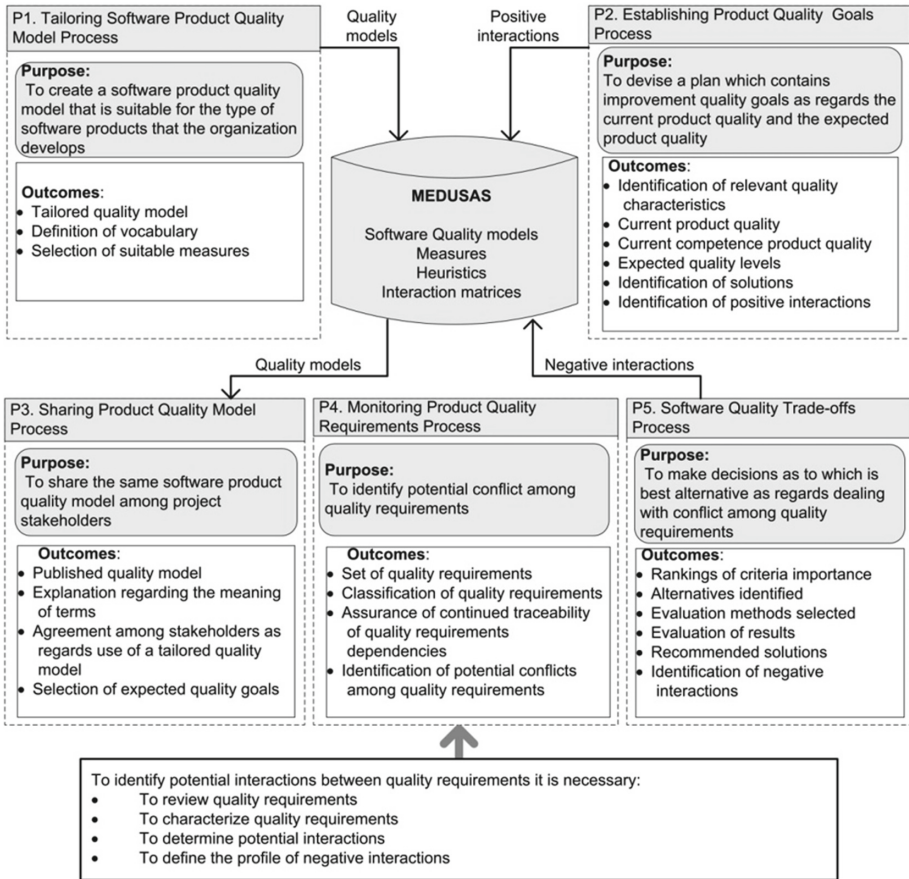


Fig. 1. SQIMF framework.

There is a need to develop methods which support the goals of eliciting and analyzing customers' quality requirements, including negotiation approaches with which to resolve conflicting interactions [5]. In industrial practice, software organizations that overlook conflicting interactions between quality requirements may confront issues related to increasing development costs and decreasing stakeholder satisfaction [3, 6]. The poor management of interactions between quality requirements could also be considered a causal factor in the failure of some projects [7–9].

Several methods with which to deal with interactions between quality requirements have been proposed, particularly negotiation approaches and prioritization methods [10]. However, the goals of these approaches and their application scope cover only specific processes as those described in ISO/IEC 12207 [10]. Interactions between quality requirements are relevant to other stages of the software development life-cycle, such as software architecture and software testing [2]. Software organizations therefore require methodological support in order to address the identification and documentation of interactions between quality requirements and the resolution of negative ones. This support should cover all the stages of the software development life cycle.

In order to provide a possible solution to the issue of managing interactions between quality requirements throughout the software development life cycle, we take into account two essential ideas; first, that the software process influences the quality of a software product, and second, that a product quality model, such as ISO/IEC 25010, can be used to generalize the quality requirements through the different processes of software development (Table 1 depicts the main quality terms used in this paper). We implemented the first idea by reviewing several process models and the literature concerning improvement initiatives. As a result, we found that few process models explicitly address quality characteristics [11, 12] and that they mention quality characteristics in processes related to eliciting and analyzing quality requirements [12]. The second idea was based on the ISO/IEC 25010, since it can be used to specify, measure and evaluate software product quality throughout the stages of the software development life cycle.

**Table 1.** Definition of some quality-related terms.

Term	Definition
Quality requirement	A requirement that a quality attribute which is present in software [13]
Quality characteristic	Category of software quality attributes that have a bearing on software quality [13]
Quality model	Defined set of characteristics, and the relationships between them, that provides a framework in which to specify quality requirements and evaluate quality [13]
Target quality goals	A description of relevant quality characteristics and their respective expected values that an organization is attempting to attain in a software product
Interaction model	A matrix-based description of interactions between quality characteristics that shows the influences of one quality characteristic on the others
Attribute	Inherent property or characteristic of an entity that can be distinguished quantitatively or qualitatively by humans or by automated means [13]

The main goal of this research, which uses process support as a basis, is to provide a process that can be used to monitor interactions between quality requirements when considering the quality characteristics described in ISO/IEC 25010. This process is part of the Software Quality Interaction Management Framework (SQIMF) (Fig. 1) which was presented in Garcia-Mireles et al. [14]. In this paper we detailed the process ‘P4.

Monitoring product quality requirements’ (P4 process in Fig. 1) and described the approach used to identify interactions between quality characteristics. The activity ‘A2. Check potential interactions between quality requirements’, which is a part of this process, was validated by conducting an exploratory case study. As an additional proposal, we included a process with which to carry out trade-offs when negative interactions occur (P5 in Fig. 1).

The paper has seven sections. Section 2 shows an overview of the main approaches used to deal with interactions between quality requirements and depicts an overview of the SQIMF. The process with which to manage interactions between quality requirements is presented in Sect. 3 while Sect. 4 presents both the exploratory case study design and its main outcomes. Section 5 presents a discussion of the results and threats to validity, while a summary of the process used to resolve conflicting interactions is provided in Sect. 6. Finally, our conclusions and future work are addressed in Sect. 7.

## 2 Related Work

Software requirements analysis and requirements negotiation are the main activities within the analysis stage of software development process during which stakeholders may discover conflicting interactions between quality requirements. According to Dahlstedt and Persson’s notion of interaction [6], the conflicting interactions between quality requirements occur when one quality requirement constrains the design or coding options of another quality requirement. In general terms, the conflicting interactions are resolved by employing trade-off methods.

Barney et al. [15] carried out a mapping study in order to identify approaches with which to perform trade-offs. They found a variety of methods, such as the Analytical Hierarchy Process, the Architectural Trade-off Method, Quality Function Deployment and algorithmic approaches, among others. As a conclusion, they pointed out that the field is immature and more research is needed to address software quality tradeoffs. The articles that [15] had categorized into both the requirements and process stage were then reviewed in order to classify them as regards their goals and the main process that the methods contained therein support. As a result, the methods were classified as either prioritization approaches if they seek only to assign a weight to each quality characteristic or negotiation approaches when the method provides a means that stakeholders can use to discuss their alternatives [10]. In addition, we found that methods with which to carry out trade-offs can be used in several processes, including those related to quality assurance.

Several methods are based on the modeling approach. For instance, the Non-Functional Requirements Framework [16] considers requirements to be goals, and more particularly quality requirements to be softgoals. Software developers should build a graph in order to describe the potential interactions between goals and the extent to which design mechanisms and components contribute to achieving those goals. Other researchers rely on both ontologies and literature surveys to model interactions between quality characteristics. The catalog of conflicting interactions can be used to identify potential interactions in new software projects [17, 18].

Our proposal for dealing with interactions is process-based. In order for a software process to contribute toward improving product quality, it should include appropriate practices [19]. Traditional software process models (such as CMMI [20], or ISO/IEC 12207 [21]) currently lack the appropriate support needed to improve product quality when it is assessed with a product quality model [12]. Indeed, there is a lack of mechanisms with which to integrate the methods required to support quality characteristics [22]. These facts and the need to support the management of interactions between quality requirements are some of the reasons why we were motivated to develop the SQIMF framework [14].

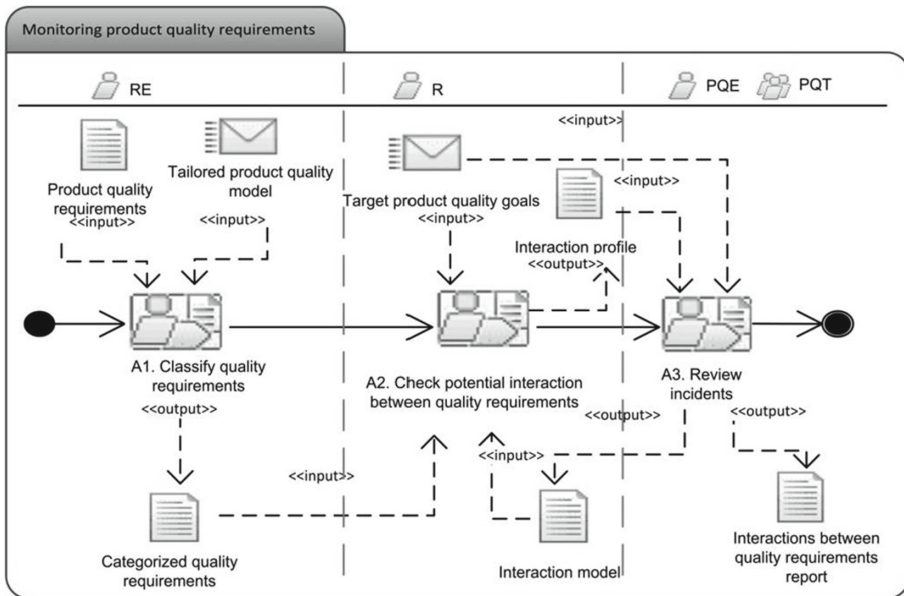
The SQIMF framework provides a set of five processes that together contribute toward identifying and documenting interactions in addition to resolving the conflicting interactions which occur during the software development process [14]. The ISO/IEC 25010 [13] product quality model was used to derive specific quality models for usability, maintainability and security. These specific models were then used to identify potential interactions between quality characteristics. The interactions identified in literature surveys were documented in interactions models, which describe the kind of relationship between quality characteristics (e.g. positive, negative, and independent) [23]. The SQIMF framework also includes a process that software organizations can use to review the practices that may be included in a particular software project in order to improve a product quality characteristic. These types of proposals are based on a mapping between practices targeted toward improving a particular quality characteristic and a software process model (please see, for example, [24]).

The processes included in the SQIMF framework can be applied at both project and organization level. At organizational level, there are two processes whose respective main goals are tailoring the product quality model to the settings in which software organizations develops software and developing an improvement initiative in order to introduce practices with which to enhance the desired product quality characteristics. At project level, three processes are aimed at: promoting a strategy which ensures that all project team members understand the quality terms, seeking interactions between quality requirements and resolving negative interactions through the use of trade-off studies. However, the processes are described only in terms of purpose and outcomes [14]. In this paper we describe two processes related to the identification of interactions between quality characteristics (P4. Monitoring interactions between quality requirements process) and the resolution of negative interactions (P5. Software quality trade-offs process). The first process (P4) also includes the design and outcomes of an exploratory case study.

### 3 Process for Monitoring Interaction Between Quality Requirements

We use the SPEM 2.0 notation [25] and the EPF composer version 1.5 (<https://eclipse.org/epf/>) to describe the process employed to monitor interactions between quality requirements (Fig. 2). The process description includes the process objectives, work products (inputs and outputs), roles, activities and an activity diagram.

The monitoring quality requirements process relies on the interaction model to uncover potential negative interactions (or conflicts) between quality requirements. Several conditions may have a tendency to lead to the appearance of interactions between quality requirements, such as clashes among stakeholders' quality requirements, the selection (or design) of software components based on quality requirements, and strict targeted values for quality requirements. When a conflict between quality requirements is identified it should be described in the interaction profile for further analysis. The main outcome of this process is the interaction profile, but it is also possible to update quality requirements and the interaction model.



**Fig. 2.** Activity diagram used to monitor product quality requirements process.

### 3.1 Process Objectives

The objectives of the monitoring quality requirements process are the following:

- A review of the consistency of quality requirements with target quality values.
- A verification of the potential interactions between quality requirements by means of the quality characteristics.
- An update of the appropriate interaction model using the interactions discovered.

### 3.2 Inputs and Outputs

The work products required in this process are: product quality requirements and related product components, a tailored product quality model, target product quality

goals, and an interaction model. The output artifacts are: a prioritized list of quality requirements, an interaction profile and a summative report containing the interactions found.

### 3.3 Roles

The roles participating in this process are presented in Table 2.

**Table 2.** Roles participating in the process.

Role name	Description
Product Quality Team (PQT)	A group of participants who have a diversity of quality interests in a particular software product. They can describe quality goals and apply appropriate methods to introduce and assess product quality
Product Quality Expert (PQE)	A participant who has the knowledge needed to adapt a product quality model in the context of organizational needs
Requirements Engineer (RE)	A participant responsible for eliciting, analyzing, specifying and validating requirements who can also categorize quality requirements using a product quality model
Reviewer (R)	A role responsible for detecting potential interactions between categorized quality requirements which can also create an interaction profile when a negative interaction is discovered

### 3.4 Activities

There are three main activities in this process: classify quality requirements, check potential interaction between quality requirements and review incidents. While the project is in progress, the activities classify quality requirements and check potential interactions that can be carried out when the process allows review sessions. We suggest that the last activity be performed when the project is at the closing stage in order to evaluate the impact of the conflicting interactions and the degree to which alternatives have resolved the problem.

- A1. Classify quality requirements activity. With regard to the tailored quality model, the RE classifies the product quality requirements. If each review session addresses changes that must be made as regards quality requirements, this activity needs to be carried out in order to update the classification of quality requirements.
- A2. Check potential interactions between quality requirements activity. Categorizing quality requirements in their respective quality characteristic allows the reviewer to identify potential interactions between quality requirements since the interaction model includes data about interactions between quality characteristics. If an interaction is identified, the Reviewer therefore needs to describe it in the interaction profile template.
- A3. Review incidents. PQE and PQT review the interaction profile reported in the software project. The analysis focuses on the characteristics of the reported



**Table 3.** Terms related to contextual factors.

Term	Description
Contextual factor	An aspect from the environment that influences either the way in which software is developed or the resulting software product
Contextual facet	A coherent set of contextual factors
Product facet	This includes contextual factors such as maturity, quality, size, system type, customization and programming language [26]
Process facet	This describes the work-flow of the development. It includes activities, work-flows, and artifacts [26]
People facet	This includes aspects related to project participants' skills and experience in addition to the assigned (project/organization) positions' jobs and roles
Organizational facet	This includes the organizational structure, organizational unit, certification, and distribution [26]
Market facet	This represents the customers and competitors. The market facet includes number of customers, market segments, strategy and constraints [26]

interactions in order to evaluate its impact on both the software project and the artifacts of the SQIMF framework (e.g., interaction model). The actions derived from this review are documented and stored in the organization's knowledge base.

## 4 Exploratory Case Study

### 4.1 Case Study Design

In order to validate the activity ‘A2. Identify potential interactions between quality requirements’ which is a part of the process ‘P4. Monitoring quality requirements’, we decided to conduct an exploratory case study. The purpose of the study was to understand how practitioners identify interactions between quality requirements and how they can be described. The study was conducted at a small software firm that we named Company A, and which was selected opportunistically since we needed an organization that was aware of how software quality can be implemented in software projects.

Company A is currently certified as a testing laboratory with the ISO/IEC 17025 standard. The company provides consulting services based on software process improvement initiatives or using the ISO/IEC 25010 to enhance product quality. Two people from Company A participated in the interviews.

We performed the case study by breaking down the activities into two groups. The first group of activities focused on the design of materials required to characterize the contextual factors that needed to be taken into account. The second group of activities was carried out a week after the first part of the study. Its main purpose was to identify and describe the profile of an interaction in a current project.



In the first part of the case study, we developed an interaction profile template whose goal was to characterize an interaction between quality requirements. The lists of factors employed to describe an interaction were extracted from Robinson et al. [4], while the context facets [26] were used to identify factors that contribute to the occurrence of an interaction. Table 3 describes the factors used in the interaction profile template.

The first version of the interaction profile template was reviewed by two researchers. Minor details concerning the interpretation of the factors were found, which were resolved by improving their explanations. The corrected version of the template was used to support semi-structured interviews.

The managing director and the quality leader, both of whom were employees at company A, were informed about the aims of this study and the need to record interview sessions. They agreed to an audio recording and also to filling in the templates and questionnaires. Both interviewees had been working with process improvement initiatives and enhancing product quality with ISO/IEC 25010 for more than four years. The main data source was based on their experience of working in the quality assurance field.

The data collection procedures required notes to be taken during the interview sessions. The notes were verified with audio files. The interactions between the quality characteristics identified by both interviewees were compared in order to gather suitable evidence for this research. Data triangulation was applied to data regarding contextual factors in order to identify relevant contextual factor for this company.

The second part of the exploratory case study was focused on the application of the activity 'A2. Check potential interactions between quality requirements' to a software project being developed by the company. The purpose of this was to understand the extent to which it would be feasible to use the process, including the artifacts, in industrial settings. A questionnaire was developed in order to obtain information about the feasibility of using the process.

## 4.2 Interviews Results

Company A decided to apply the process for monitoring interactions among quality requirements in a new project they were working on. The software to be delivered was a web application which supports an organization as regards providing web information content for a target audience that includes the visually impaired. The exploratory case study was conducted at the conception stage of this web project.

The interaction profile includes a section that addresses the interaction model. In this case, it was developed in order to determine the type of relationships between security and usability sub-characteristics. The interviewees used their own experience and the features of the software project under study as a basis on which to establish the type of interaction. For instance, Table 4 shows an interaction model between security and usability filled in by one of the interviewees.

Positive interactions are marked with the sign (+) while negative interactions are marked with (-). The sign (O) is used when the interviewee does not have sufficient information to ensure that there is an influence between the quality sub-characteristics

**Table 4.** Interaction model filled in by an interviewee.

Product usability →	Appropriateness recog- nizability	Learnability	Operability	User error protection	User interface aesthetics	Accessibility
Security						
Authenticity	O	O	-	O	O	-
Confidentiality	O	O	O	O	O	O
Conformance	O	O	O	O	O	O
Attack detection	O	O	O	O	O	O
Availability	O	O	O	+	O	O
Integrity	O	O	O	O	O	O
Non-repudiation	O	O	O	O	O	O
Traceability	O	O	O	O	O	O

under review. In this case study, the interviewees reported positive interactions between availability and user error protection. They also reported two negative interactions between the pairs authenticity – operability and authenticity – accessibility.

After identifying an interaction between quality requirements, the next step was to characterize the relevant factors that foster it. The interviewees used the contextual facets to report factors related to the product and process facet.

The main factors within the product facet are quality and application type. Quality refers to usability and security requirements that the project should address. The needs of a particular set of targeted users constrain both the design and implementation options of security mechanisms, since they are visually impaired people. In addition, the application type influences the security mechanisms that can be implemented. Moreover, the characteristics of the screen sizes and interaction mechanisms also need to be considered when designing the web application.

With regard to the process facet, one of the interviewees suggested that the review of software increments at the end of a software development process iteration might be an appropriate means to identify potential interactions between quality requirements. The executable version of software can be used to evaluate the quality requirements. In the light of the testing results, the customer can make decisions concerning how quality requirements were achieved. The use of this review approach allows both the software firm and its customers to negotiate negative interactions between quality requirements. Although the study was focused on identifying interactions between quality requirements, the interviewee also requested methods, tools or practices that support the management or resolution of negative interactions. He also suggested that software developers need training if they are to manage conflicting interactions.

With regard to the organizational facet, Company A works by means of projects. For this web application the team consisted of four team members. However, these factors were not relevant as regards describing an interaction. With regard to the market facet, the interviewees did not consider that any of its factors might contribute to the occurrence of an interaction. In the case of the person facet, the interviewees did not consider that people's knowledge and skills were potential influencing factors. However, since the interviewees provide consulting to improve software quality, their knowledge may be a relevant contextual factor as regards identifying interactions.

The interviewees determined a negative interaction between accessibility and authenticity. They found particularly difficult to provide access support for all types of users, including those suffering from blindness. As an argument they commented that "a common approach employed to register users in a web system is that of using CAPTCHAs, but they distort a label as regards differentiating between a real user and a bot." However, this mechanism requires an in-depth study in the context of this web application owing to the profile of intended users.

In summary, the negative interaction only occurs with a particular group of users when the application should display appropriate information (resources) for each type of user. This signifies that the quality requirements for a web application are the main contextual factors that contribute to the occurrence of a negative interaction.

The interviewees additionally highlighted positive interactions between a pair of quality characteristics. They reported that integrity (security) has a positive influence on user error protection (usability). The rationale for this relation is that the security mechanisms implemented ensure that only the user with modification access can change data records. This interaction relies on their previous experience in developing and assessing systems.

The interaction profile template was therefore a feasible instrument with which to characterize a negative interaction between quality characteristics. The interaction model serves as a guideline to determine the type of interaction between quality characteristics. The main contextual facets that were relevant for Company A in the project under study were both the product and process facets. However, the person facet needs to be studied in great depth in order to determine to what extent the participants' skills contribute toward identifying and characterizing negative interactions.

The second part of the exploratory case study was focused on studying the feasibility of using the monitoring quality requirements process in a software firm. Our main goal was to characterize the interactions between quality requirements. We developed guidelines for the use of the interaction profile template and the activity diagram. After the participant had finished the tasks in the process, we asked him to answer a questionnaire developed to understand the suitability of the process.

With regard to the template for the interaction profile, the participant was clearly able to fill in identification data: project id, date, reviewer, type of software and artifacts analyzed. With regard to the interaction model, it can serve to identify both positive and negative perceived interactions between quality sub-characteristics. When the participant filled in the factors that characterized the interaction, his responses were based mainly on his previous experiences in developing software and assessing software quality. Since the project under study was at a conception stage, this may explain

why there are few references to specific means or requirements used to describe the interaction and its potential impact. Table 5 shows paraphrased responses for relevant factors.

The analysis of the questionnaire filled in by the participant showed that process objectives, roles, descriptions and work products are clearly described. However, the process tasks and the interactions profile could be improved to support the identification of interactions. With regard to the understandability of process elements, the

**Table 5.** Responses to diverse factors used to describe an interaction profile.

Factor	Response
<i>Basis</i>	In this project, adding security requirements may have a negative influence on accessibility and operability since the software features are only available to certain types of users
Which quality requirements are involved in an interaction? Do the contextual factors have an influence on a given quality requirement?	
<i>Criterion</i>	New security components affect system structure.
Which reasons are considered to lead to these interactions?	
<i>Establish the degree of the interaction</i>	Application type and target users might impact on the degree of interaction between quality characteristics.  The stakeholder's experience in the security field may influence the quality of security requirements.  The expert's knowledge can be used to establish a security mechanism to reduce the influence of highly secure mechanisms on accessibility or operability.
What is the scope of the interaction between quality requirements? What features, components or users' categories are involved?	
<i>Probability of occurrence</i>	The interactions occur during the software development under the constraints considered.
What is the probability of a conflicting interaction occurring?	
<i>Impact of the interaction</i>	The main effect: Application does not meet basic quality requirements. Unsatisfied users and application cannot be delivered to target users.
What is the effect of the interaction on the software project? For instance: Catastrophic, inconvenient, system failure, system reboot, unsatisfied users.	
<i>Type of interaction</i>	Perceived interaction
What is the type of this interaction? It is a perceived interaction when it is described at requirements level. It is an implementation interaction when it is based on the analysis of implementation means	
<i>Context</i>	Main contextual factors: Application users and application type.
What contextual factors influence the interaction between quality characteristics?	

questionnaire answers depicted that process objectives, description of roles, work products and templates are easy of understand. The process elements of the tasks should, however, be improved.

When asked to state the extent to which process objectives are easy to apply, the participant marked the disagree option. The comments written in the instrument showed that there is a lack of information with which to understand how the interactions can be identified when software quality measures and indicators are used. With regard to this last comment, the experience of previous software quality assessment can be used to identify interactions between quality characteristics. Moreover, the quality goals should be linked to specific practices in order to evaluate whether the practices contribute toward resolving negative interactions.

## 5 Results and Discussion

The exploratory case study has provided evidence about the potential usefulness of the SQIMF. The interaction matrix and the characterization of interactions using the contextual facets were relevant as regards establishing a profile of the conflicting interactions between security and usability.

The interaction matrix was provided in order to determine the type of relationships that occur between the sub-characteristics of usability and security. The case study participants found it easy to fill in the matrix. One reason for this is that this company is focusing on evaluating the quality of software products from the process and product perspective. Although they could be considered expert practitioners in the field of software quality, this case study was the first time that they had addressed the topic of interactions between quality characteristics.

Several studies have reported interactions between software quality characteristics, including security and usability [1], but few empirical studies address the interaction issues by taking into account the sub-characteristic which belongs to each quality characteristic. This is therefore an important finding to be considered when dealing with interactions.

The purpose of the interaction profile is to characterize the interaction between quality characteristics using the contextual facets. The evidence gathered through this case study showed that the relevant factors that influence the interaction between usability and security requirements are the user's quality needs, user's profiles, and the type of application. The type of application is a factor that determines the type of quality attributes to be addressed in a software project [3, 27]. With regard to the process facet, the interviewees believed that a customer can participate in evaluating software increments. At the end of each iteration, the customer can review the software execution in order to identify any conflicts with the initially established quality requirements.

The interviewees did not consider the remaining context facets, such as people, organization, and market, to be factors that identify conflicting interactions between quality requirements. One reason for this is that the project was at its conception stage and the final set of requirements to be addressed in the software project had not as yet

been specified. The identification of interactions is therefore based on the interviewees' perceptions and the main factors were the stakeholders' quality needs and product type.

The exploratory case study has shown that the process for monitoring product quality requirements can help in the identification of interactions between quality requirements and that it can be used in the conception stage of a software project. The activity 'A2. Check potential interaction between quality requirements' is useful as regards identifying and characterizing conflicting interactions between quality requirements. However, the tasks related to this activity need some refinement for use in industrial projects. Furthermore, it is necessary to validate the other activities in this process: A1. Classify quality requirements and A3. Review incidents.

In order to mitigate the effects of threats to validity, we followed the guidelines of Runeson et al. [28]. As regards the external validity, the exploratory case study carried out cannot be generalized to other companies. Nevertheless the characterization of the organization and the outcomes related to conflicting interactions between usability and security can provide useful insights into the development of a theory with which to characterize interactions at the conception stage of a software product. Moreover, we identified interactions between quality sub-characteristics that can be generalized to their respective quality characteristics considering the hierarchical structure of the quality models. This result is thus consistent with reports of conflicting interactions between usability and security [1, 8].

The research was kept under control through the application of the template approach, because it allows the design of instruments and the a priori establishment of how the instruments can contribute to the research [28]. Furthermore, in order to improve the reliability of the artifacts designed, all of them were checked by two researchers. Moreover, we used findings obtained from different sources to apply data triangulation when identifying evidence [28].

Since this was the first time that Company A had worked with interactions between quality requirements, the first part of the case study addressed the contextual factors. These were commented on with the interviewee in order to clarify the terms.

With this action, we thus attempted to mitigate the effects of construct validity. In addition, the guidelines used in the case study included descriptions in order to support participants when filling in the templates. However, it was not possible to interview the participants about whether these materials were useful as regards identifying interactions between quality characteristics.

## 6 Towards Resolving Negative Interactions Between Quality Characteristics

We propose that the software quality trade-off process (Fig. 3) can be used to resolve conflicting interactions between quality characteristics. The purpose of the software quality trade-off process is to make decisions concerning the best resolution alternative when conflicting interactions between quality requirements appear during software development. The process employs a rationalistic approach based on criteria defined in order to evaluate the potential resolution alternatives, and also considers the appropriate

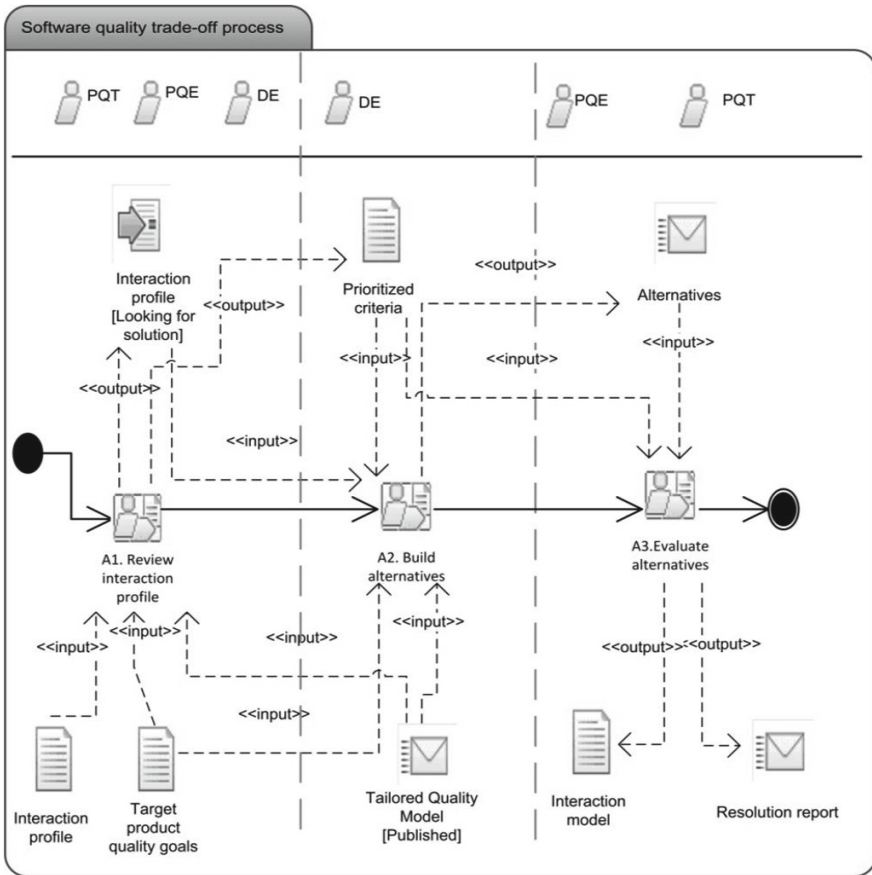


Fig. 3. Activity diagram for software quality trade-offs process.

methods with which to analyze the alternatives used to resolve conflicting interactions between quality characteristics.

The process starts when an interaction profile is created and the software development team wishes to resolve the conflicting interaction that has been discovered. The main outcome of executing this process is a set of solution alternatives that are then evaluated in order to recommend that which satisfies the decision criteria. Furthermore, as a result of applying this process the interaction model can be updated.

## 6.1 Process Objectives

The software quality trade-off process can be used to attain the following:

- A systematic approach can be applied to analyze conflicting interactions and to provide a recommended solution.



- The set of alternatives can be built using appropriate methods.
- Both the recommended solution and the interaction profile can be used to update the interaction model.

## 6.2 Inputs and Outputs

The work products required in this process are: tailored product quality model (published), target product quality goals and interaction profile. Moreover, the interaction model is an input to be updated during the implementation of this process. The process outcomes are the following work products: the decision to carry out a further analysis of the interaction profile, the prioritized criteria used to assess alternatives, the set of alternatives and the resolution report. The interaction profile can additionally be updated during the implementation of this process.

## 6.3 Roles

Table 6 depicts the roles that participate in this process.

**Table 6.** Roles participating in the software quality trade-off process.

Role	Description
Product Quality Team (PQT)	Described in Table 2.
Product Quality Expert (PQE)	Described in Table 2.
Discipline Expert (DE)	This role has the knowledge and skills needed to deal with product quality characteristics in the context of the process under study. The role's responsibilities include the construction of alternatives with which to resolve a conflicting interaction in addition to selecting and applying methods with which to analyze proposed alternatives.

## 6.4 Activities

The process consists of three activities: review interaction profile, build alternatives and evaluate alternatives. These are described in the following paragraphs.

- A1. Review interaction profile. The PQE and the PQT review the interaction profile in order to determine further activities as regards conflicting interactions between quality requirements. The review is enriched with the DE's opinions with regard to the specific quality characteristics and process under study. The outcome of the review is a decision concerning the relevance of additional activities in order to recommend a solution.
- A2. Build alternatives. The DE builds a set of potential alternatives with which to resolve the interactions that have arisen between quality requirements in the context

of the software process in which they emerge. The methods used to assess alternatives are additionally documented and used to determine the extent to which each alternative achieves the assessment criteria previously established.

- A3. Evaluate alternatives. The PQE and the PQT roles review alternatives and the results of the assessment methods. The alternatives are assessed by considering the assessment quality criteria and procedures established to carry out the assessment. The resolution report should include the means used to resolve the conflicting interaction and how the selected means could be implemented in the software project. The interaction model is also updated with the interaction profile information and with the recommended solution.

## 7 Conclusions

Software organizations need appropriate support to manage interactions between quality requirements. In order to support them, in this paper we have described a process with which to monitor interactions between quality requirements. An exploratory case study has also been conducted to validate the activity ‘A2. Check potential interactions between quality requirements’. This resulted in the main contextual factors that contribute to the identification and characterization of an interaction. Furthermore, we have presented a summary of a process that can be used to resolve conflicting interactions.

Although we prepared a template and guideline to support the exploratory case study participants as regards understanding interactions and contextual factors, we found that interactions between quality requirements were reported as a perceived interaction because the project under study was at the conception stage. However, without explicit reference to the potential impact of the interaction on the software project it is difficult to understand to what extent it may influence software development activities or project success. The identification of perceived interaction is a first step toward understanding and characterizing an objective interaction, but it is necessary to include other approaches in order to uncover the real impact of the interaction on the software project. For instance, a risk-based technique would provide information about the impact of the interactions.

With regard to the validation of the monitoring product quality requirements process, the results showed that it can be used for a software organization which deals with product quality, but some tasks should be improved. With regard to the artifacts used in the exploratory case study, the participants stated that the interaction profile is easy to use and apply. They also suggested taking into account indicators and measures of software product quality in order to establish a mechanism with which to identify interactions between quality characteristics.

As future work, it will be necessary to validate the process used to monitor interactions between quality characteristics in other stages of the software development life cycle and also to consider both other organizational contexts and application domains. A software tool currently provides information about interactions between quality characteristics, but it needs to be improved so as to address the information used

to characterize an interaction. The process employed to resolve negative interactions should also be validated by means of empirical studies.

**Acknowledgements.** This work has been funded by the VILMA and INGENIOSO projects (Consejería de Educación, Ciencia y Cultura - Junta de Comunidades de Castilla La Mancha) and Fondo Europeo de Desarrollo Regional FEDER, Ref.: PEII11-0316-2878 and Ref. PEII11-0025-9533) and GEODAS-BC project (TIN2012-37493-C03-01 funded by the Spanish Ministerio de Economía y Competitividad and by FEDER (Fondo Europeo de Desarrollo Regional).

## References

1. Mairiza, D., Zowghi, D., Nurmiani, N.: Towards a catalogue of conflicts among non-functional requirements. In: 5th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2010, pp. 20–29 (2010)
2. Ameller, D., Ayala, C., Cabot, J., Franch, X.: Non-functional requirements in architectural decision making. *IEEE Softw.* **30**, 61–67 (2013)
3. Chen, L., Babar, M.A., Nuseibeh, B.: Characterizing architecturally significant requirements. *IEEE Softw.* **30**, 38–45 (2013)
4. Robinson, W.N., Pawlowski, S.D., Volkov, V.: Requirements interaction management. *ACM Comput. Surv.* **35**, 132–190 (2003)
5. Loucopoulos, P., Sun, J., Zhao, L., Heidari, F.: A systematic classification and analysis of NFRs. In: 19th Americas Conference on Information Systems, AMCIS 2013 - Hyperconnected World: Anything, Anywhere, Anytime, pp. 208–217, Chicago, IL, USA (2013)
6. Dahlstedt, A., Persson, A.: Requirements interdependencies: state of the art and future challenges. In: Aurum, A., Wohlin, C. (eds.) *Requirements engineering*, pp. 95–116. Springer, Berlin Heidelberg (2005)
7. Boehm, B., In, H.: Identifying quality-requirement conflicts. *IEEE Softw.* **13**, 25–36 (1996)
8. Theofanos, M.F., Pfleeger, S.L.: Guest Editors' introduction: shouldn't all security be usable? *IEEE Secur. Priv.* **9**, 12–17 (2011)
9. Thakurta, R.: A framework for prioritization of quality requirements for inclusion in a software project. *Softw. Qual. J.* **21**, 573–597 (2013)
10. García-Mireles, G.A., Moraga, M.Á., García, F., Piattini, M.: Methods for supporting management of interactions between quality characteristics. In: Filipe, J., Maciaszek, L. (Eds.) 9th International Conference on Evaluation of Novel Approaches to Software Engineering, pp. 93–100. INSTICC, Lisboa (2014)
11. Unterkalmsteiner, M., et al.: Evaluation and measurement of software process improvement—a systematic literature review. *IEEE Trans. Softw. Eng.* **38**, 398–424 (2012)
12. García-Mireles, G.A., Moraga, M.Á., García, F., Piattini, M.: Towards the harmonization of process and product oriented software quality approaches. In: Winkler, D., O'Connor, R.V., Messnarz, R. (eds.) *EuroSPI 2012. CCIS*, vol. 301, pp. 133–144. Springer, Heidelberg (2012)
13. ISO, ISO/IEC FCD 25010: Systems and software engineering - system and software product quality requirements and evaluation (SQauRE) - System and software quality models (2010)

14. García-Mireles, G.A., Moraga, M.Á., García, F., Piattini, M.: A framework to support software quality trade-offs from a process-based perspective. In: McCaffery, F., O'Connor, R.V., Messnarz, R. (eds.) *EuroSPI 2013. CCIS*, vol. 364, pp. 96–107. Springer, Heidelberg (2013)
15. Barney, S., Petersen, K., Svahnberg, M., Aurum, A., Barney, H.: Software quality trade-offs: a systematic map. *Inf. Softw. Technol.* **54**, 651–662 (2012)
16. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publisher, Dordrecht (2000)
17. Al Balushi, T.H., Sampaio, P.R.F., Loucopoulos, P.: Eliciting and prioritizing quality requirements supported by ontologies: a case study using the ElicitO framework and tool. *Expert Syst.* **30**, 129–151 (2013)
18. Mairiza, D., Zowghi, D.: An ontological framework to manage the relative conflicts between security and usability requirements. In: 3rd international workshop on managing requirements knowledge, MaRK2010, pp. 1–6 (2010)
19. Allen, J., Kitchenham, B., Konrad, M.: Theme Q. The relationships between processes and product qualities. In: Forrester, E., (ed.). vol. pp. 19–28. Software Engineering Institute, Carnegie Mellon. (2006)
20. CMMI, P.T. CMMI for Development, Version 1.3 (CMU/SEI-2010-TR-033) (2010). cited 2012, <http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm>
21. ISO: ISO/IEC 12207 Systems and software engineering — Software life cycle processes (2008)
22. Chiam, Y.K., Staples, M., Ye, X., Zhu, L.: Applying a selection method to choose Quality Attribute Techniques. *Inf. Softw. Technol.* **55**, 1419–1436 (2013)
23. García-Mireles, G.A., Moraga, M.Á., García, F., Piattini, M.: Identificación de interacciones entre las características de calidad del software. In: XVIII Jornadas de Ingeniería del Software y Bases de Datos JISBD2013, pp. 141–154. Universidad Complutense de Madrid, Madrid, España (2013)
24. García-Mireles, G.A., Moraga, M.Á., García, F., Piattini, M.: The influence of process quality on product usability: a systematic review. *CLEI Electron. J.* **16**, 1–13 (2013). <http://www.clei.org/cleij/paper.php?id=278>
25. OMG, Software & Systems Process Engineering Metamodel specification (SPEM) Version 2.0 (2008)
26. Petersen, K., Wohlin, C.: Context in industrial software engineering research. In: 3rd International Symposium on Empirical Software Engineering and Measurement ESEM 2009., IEEE, Editor, pp. 401–404. Lake Buena Vista, FL, USA (2009)
27. Berntsson Svensson, R., et al.: Quality requirements in industrial practice-an extended interview study at eleven companies. *IEEE Trans. Softw. Eng.* **38**, 923–935 (2012)
28. Runeson, P., Höst, M., Rainer, A., Regnell, B.: *Case Study Research in Software Engineering: Guidelines and Examples. Case Study Research in Software Engineering: Guidelines and Examples*. John Wiley and Sons (2012)

Evaluation of Novel Approaches to Software  
Engineering

10th International Conference, ENASE 2015, Barcelona,  
Spain, April 29-30, 2015, Revised Selected Papers

Maciaszek, L.A.; Filipe, J. (Eds.)

2016, XI, 195 p. 38 illus. in color., Softcover

ISBN: 978-3-319-30242-3