TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABREVIATIONS AND ACRONYMS

NFRs	Non-functional Requirements
FRs	Functional Requirements
AOs	Architectural options
BG	Business goal
RBG	Refined business goal
SQE	Software quality Engineering
QRs	Quality requirements
CQL	Consolidated Quality Life cycle
IQUAL	Ingénierie de QUAlité Logicielle
SQuaRE	Software Product Quality Requirements and Evaluation
SPACE-UFO	Software Product Advanced Certification and Evaluation-User Focus
SQUID	Software QUality In Development
Prometheus	Probabilistic Method for early evaluation of NFRs
BBNs	Bayesian Belief Network
GQM	Goal Question Metric
SQUARE	System QUAlity Requirements Engineering
SHEL	Software HardwarE Live ware
KA	Knowledge Area
SWEBOK	SoftWarE Body Of Knowledge
CQL	Consolidated quality life cycle
QTM	**Q**uality **T**raceability **M**odel

UP	**U**nified **P**rocess
QiU	**Q**uality **i**n **U**se
EQ	**E**xternal **Q**uality
IQ	**I**nternal **Q**uality
ATAM	**A**rchitecture **T**radeoffs **A**nalysis **M**ethod
FDAF	**F**ormal **D**esign and **A**nalysis **F**ramework
MOQARE	**M**isuse **O**riented **Q**u**A**lity **R**equirements **E**ngineering
SOQUAREM	SOtware QUAlity Requirements Engineering Method
QAs	**Q**uality **A**ttribute**s**
QM	Quality Model
FSS	**F**ield **S**ystem **S**imulator
MSLite	Management Station
SOP	Standard **O**perating **P**rocedure
IESE NFR	**I**nstitute for **E**xperimental **S**oftware **E**ngineering **N**on **F**unctional **R**equirements
IT	Information Technology
SHEL	**S**oftware **H**ardwar**E** and **L**ive ware

INTRODUCTION

PRESENTATION OF THE RESEARCH PROJECT

A. Subject of the research project

Software requirements engineering is a large and complex discipline requiring more and more expertise and knowledge from practitioners and software developers. With the rapid evolution in the field of software development and the increasing pressure to deliver high quality applications, this discipline is faced with major problems such as: a) lack of systematic guidance on how to elicit quality requirements (called also Non Functional Requirements NFRs); b) difficulty identifying quality requirements and representing them in models and processes and c) absence of clear guidelines about the way to provide a consensus view on quality characteristics and their relationships. The existing techniques of requirements capture (as viewpoint and object-oriented) do not put emphasis on quality requirements as is the case for functional requirements (Araujo et al., 2003). Experience shows that approximately 70 percent of software projects have failed to deliver what originally was required. Consequently, developed applications are often costly in terms of resources and time and the estimated cost per defect increases significantly in the latter stages of the software development life cycle. Furthermore, they rarely respect time deadlines and are often returned by dissatisfied users (NIST, 2002 and Humphrey, 1995). ;

On the other hand, recent studies (Sommerville et al., 1997), (Cysneiros et al., 2004), (Bredemeyer et al., 2001), (Mylopoulos et al., 1992), (Hill et al., 2004), (Wiegers et al., 1999) and (Poort et al., 2004) demonstrate that the quality requirements specification step is ignored or bypassed for various reasons: quality is considered as an afterthought, cost and/or absence of quality engineering practices. The lack of this step in the definition phase of the software product life cycle may compromise business processes and may impact negatively the results of any development project.

QRs management of the software product is an emerging discipline aiming to palliate these

problems and develop high quality software systems. New QRs management approaches have been developed to specify and model NFRs at the early stages of the life cycle. They used QAs as force drivers to evaluate architectures and make early identification of risks, sensitive points and tradeoffs before design decisions are made (Gallagher, 2000). They also used NFRS as quality aspects to evaluate architecture designs and to predict early design errors and be able to improve them before delving into implementation features (Dai et al., 2005). The next section will situate the research project between the traditional software engineering approaches and current software quality engineering standards.

B. Context of this research project

The context of this reserach project is related to the management of QRs at early stages of the software product life cycle. As illustrated in Figure 1.1, this research project is situated between traditional software engineering approaches and existing quality standards.

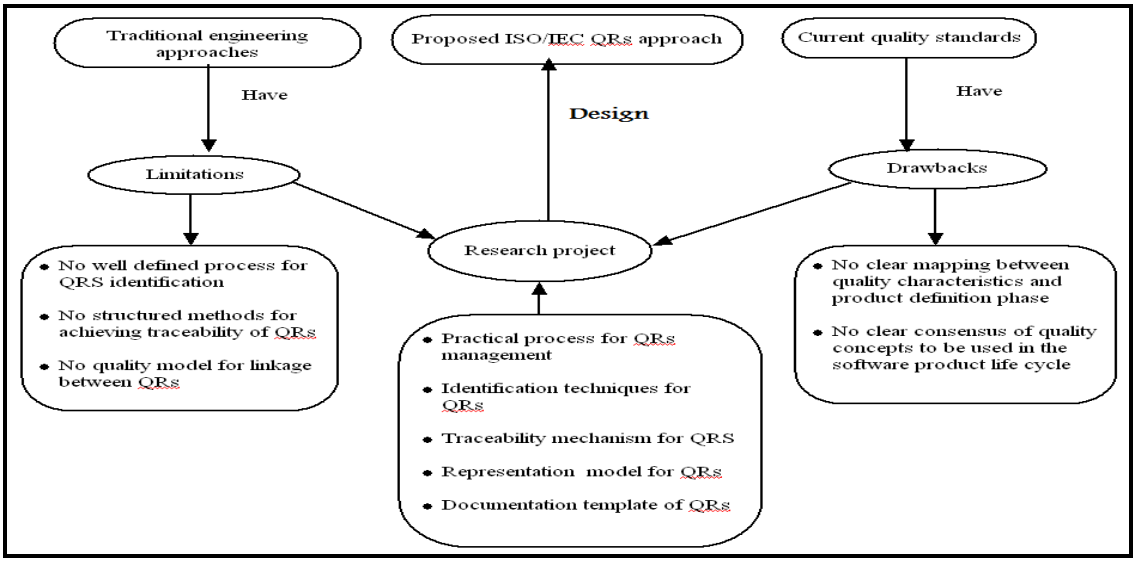


Figure 1 Context of the research project

The traditional software engineering approaches (like viewpoint and object oriented methods) are proving to improve the quality of requirements specification (Kotony and Sommerville., 1996). Viewpoint-oriented approachs support both the requirements elicitation

and the structuring of the requirements document. They enable the conversion of top-level goals into requirements and constraints. PREview (Process and Requirements Viewpoints) is a requirements method focusing on the early stage of requirements engineering (Sawyer et al., 1996).

But these approaches are faced with the following limitations:

- Quality requirements identification step is considered as an afterthought;
- No clear guidance is provided to identify and define QRs;
- No well structured process for QRs identification;
- No defined methods for retracing QRs;
- No defined quality model for dynamic linkage between QRs;
- Lack of a consensus on the definition of QAs.

On the other hand, software quality engineering standards have proven their usefulness in different fields of application such as facilitation of communication between users through a standard vocabulary (ISO/IEC 9126, 2004) and (ISO/IEC 14598, 1999). However, one notes that the emergence of software quality engineering standards in the development of software product systems has not solved some of the problems associated with the software QRs management.

This research project addresses some of the limitations that existing software engineering approaches and software quality engineering standards suffer from in order to design the ISO/IEC standards-based quality approach (Figure1.1). For example, this research project addresses the limitation: “No well defined process for identification of quality requirements” and the drawback: “No clear consensus of quality concepts” by proposing a “software QRs management process”.

C. Contribution and foreseen benefits of this research project

As mentioned in the previous section, this research project will address some of the limitations from which existing engineering approaches and software quality engineering

standards suffer. The relevant added value of this research can be described as follows:

- The research solution will be proposed to the editor committee of the guide to SWEBOK (SWEBOK, 2004) for the consideration;
- The research solution will be given for the disposition of the international standardization (ISO SC7 / WG6) once published;
- The research solution will provide the software industry a structured QRs engineering method that can be used to support requirements engineering phase.

Figures 2 and 3 summarize the difference between the traditional software engineering process and the new enhanced process with quality concepts. The traditional functional requirements definition process in the specification phase seeks maximum or even all requirements defined or frozen. In practice these requirements are often modified or even sought for in further phases of the life cycle (Figure 2), while quality requirements may be partial and require further elicitation, definition and refinement during the development process. Figure 3 shows the different categories of software QRs identified at each phase of the development process. They are described in (Suryn, 2003) where QRs are extracted from the stakeholder's requirements and translated through the decomposition model into the three categories of ISO/IEC 9126: internal and external quality requirements (IQ and EQ), quality in use (QiU) and the operational quality (Oper) of the TL 9000 standards.

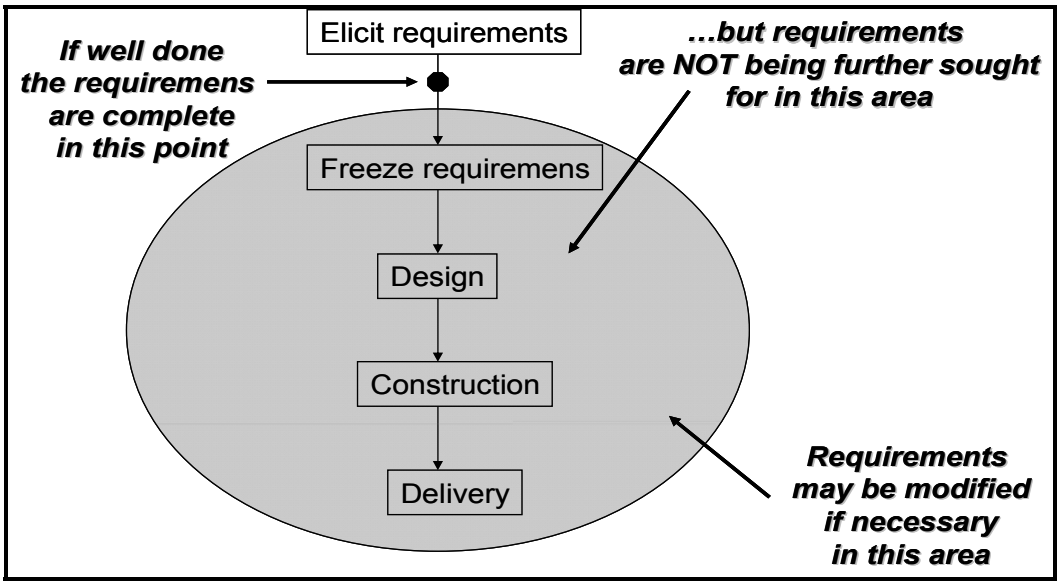


Figure 2 Traditional functional requirements in software engineering process
Extracted from Suryn (2006)

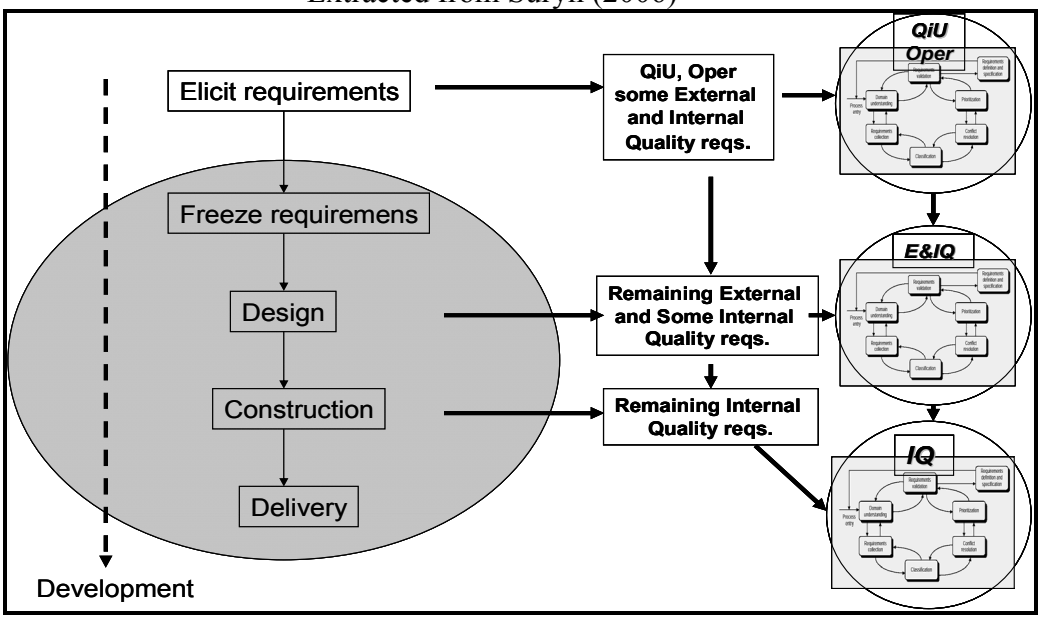


Figure 3 Quality requirements in software engineering process
Extracted from Suryn (2006)

The dashed oriented arrow indicates that quality requirements could be clarified and refined from elicited requirements to the construction phase. At each development phase, they could be formalized according to the traditional requirements formalization process where requirements are analyzed, collected, classified and prioritized to finally be validated.

Further, as shown in Figure 3, quality in use (QiU) and operational quality (OP) are the first categories of software QRs that could be identified at the requirements phase. Some external (EQ) and internal quality (IQ) requirements could also be defined. At the design phase, the external and some internal QRs could be defined while at the construction phase, only the internal QRs are identified.

Among the challenging problems addressed in this thesis are the following:

(a) Investigating various aspects of software QRs management such as identification (of business and software) requirements, specification, representation and documentation;
(b) Supporting this management by software quality engineering standards.

The key motivations for this research project are:

- The need to map quality concepts with the product definition phase;
- The need to support both novices and experts in software QRs management.

D. Research objectives

This research goal can be stated as follows: "Support the software product definition phase with a management method of quality requirements: identification, representation and documentation".

To pursue this goal, the research objectives are to:

1. Develop a structured quality requirements engineering method: **SO**ftware Product **QUA**lity **R**equirements **E**ngineering **M**ethod (SOQUAREM). The quality standard ISO/IEC SQuaRE 25030 is used as a framework supporting the engineering process of the method.

2. Develop the process model representing concepts and phases of SOQUAREM method.

The research methodology adopted to achieve these objectives is divided into four main phases: exploration, analysis, design and application.

The exploration phase consists of exploring the software quality requirements domain, specifically the software quality concepts, QRs definitions and quality standards (ISO/IEC 9126 and ISO/IEC 25030). This phase also describes and analyzes the existing QRs management approaches and establishes their strengths and weaknesses.

The analysis phase is divided into three sub phases: Analysis of existing software QRs management methods, data collection of software QRs engineering practices in industry and analysis of resulted indicators from industrial and academic environments.
The first sub phase consists of analyzing some representative software QRs management methods (chosen from literature review) in their case studies to know to what extent they address management of QRs. The approach adopted during this analysis is to describe the applicability of these methods by analyzing their case studies in the applicative domains and identifying their strong and weak points in industrial and scientific communities.
The second sub phase consists of analyzing collected data from a questionnaire in industry to determine the current state of the QRs engineering practices.
The third sub phase analyzes industrial and academic indicators obtained in the two preceding sub phases and identifies critical needs seen by industry in the field of software QRs management. Important conclusions and justifications of the proposed solution are formulated.

The design phase consists of creating the software QRs engineering method (SOQUAREM) and the associated process model. The main concepts involved in the engineering process of the method (BMM and BCT, scenarios template, utility tree and QAs template) are

developed and detailed. The main phases of the SOQUAREM process model are then described. The phases are:

1. State the business goals;
2. Refine the business goals;
3. Link the refined business goals to quality attributes;
4. Build quality attributes scenarios;
5. Consolidate quality attributes;
6. Link quality attributes to the functional process.

The application phase of the method consists of applying the method in an illustrative example for a building automation system to clarify the core ideas of SOQUAREM. The method is then evaluated in industry (by international experts in the software quality field) and academia (during workshop sessions) and on a committee level ISO/IEC SC7 System and Software Engineering. Feedback on SOQUAREM is provided by both experts and participants of the workshop session and are analyzed for further improvements and future research avenues.

E. Limitations of the research

The present research is limited to the design of the SOQUAREM method and its process model for managing software QRs and does not cover implementation of IT tool supporting this process. It is important to notice that linkage of measures with the associated quality characteristics is not part of this research project because the measures in ISO/IEC 2500 are not available. On the other hand, as the process is involving stakeholders at each phase (during the consensus session to discuss and confirm QAs), the used negotiation techniques were not investigated when they do not approve the resulted QAs. One supposes that negotiation is done and the required phase is restarted. In addition, the developed questionnaire is not deployed in a large industrial spectrum due to time constraints and availability of respondents. The questionnaire has been deployed with eight domain representatives of industry who accept to distribute it in their respective companies. The main

purpose of the questionnaire is to have some indications about QRs engineering practices in industry. On the other hand, the conflicts resolution among QAs and their prioritization is a vast and complex subject which could not be entirely treated as part of this thesis. Fictitious data is provided to illustrate the conflicts resolution problem. Finally, evaluation of the whole process by standards or methods could not be performed under the mandate of this thesis, being limited to analyzing feedback from software quality domain experts and participants of the workshop session (Only the first four phases of the process have been evaluated during the workshop session).

F. Organization of this Thesis

The organization of this thesis is as follows:

The chapter 1 presents the literature review on the main concepts and definitions related to software QRs management, in particular, the quality requirements and software quality definitions, software quality engineering standards and existing software QRs management methods such as Soft Goal Notation, MOQARE (**M**isuse **O**riented **Q**u**A**lity **RE**quirements), IESE NFR (**I**nstitute for **E**xperimental **S**oftware **E**ngineering **N**on **F**unctional **R**equirements) method, FDAF **(F**ormal **D**escription and **A**nalysis **F**ramework) and ATAM (**A**rchitecture **T**radeOff **A**nalysis **M**ethod).

The chapter 2 presents research objectives and the research methodology designed to address this research. Research steps to accomplish the stated objectives are also described in detail.

The chapter 3 explains the details of the research execution and gives the justifications of the research solution. Interest is centered on QRs indicators of both academic and industrial environments. First, applicability of chosen quality requirements management methods from literature review is analyzed and discussed. Analysis is based primarily on established strengths and weakness of existing methods and quality requirements engineering criteria. Second, the current situation of quality requirements environment is analyzed in industry. A questionnaire is elaborated for this purpose. Third, resulted indicators from the two

environments are analysed and discussed and important obseravtions are revealed to finally formulate requirements for the proposed research solution. An overview of the proposed research solution and its innovative aspects are presented by describing its specific features, meta-model, building process and process structure.

The chapter 4 describes in detail the proposed research solution **called SO**ftware **QUA**lity **RE**quirements **M**Ethod (***SOQUAREM***) and includes key concepts and an elaborated SOQUAREM process model.

The chapter 5 describes the application of the SOQUAREM process to an automation building system by an illustrative example. Finally, the process is analyzed and discussed and its practical relevance is evaluated.

This work then summarizes the key contributions, implications for software engineering, practical implications, limitations, strengths and future research avenues.

CHAPTER 1

LITERATURE REVIEW

1.1 Introduction

This chapter presents the literature review of existing quality requirements definitions, quality requirements management methods and quality standards for a software product. Section 1 introduces a quality requirements concept and related terminologies. Important definitions of quality needs, quality requirements and software quality as seen by major actors and (SWEBOK) are presented. Section 2 describes concepts of quality standards. The third section defines and investigates concepts of various quality requirements management methods designed at different levels of software development (requirements and architecture levels) (MOQARE, IESE NFR, Soft goal notation, ATAM, Prometheus and quality attributes model). Section 4 presents a comparative analysis of these designed methods and establishes their weakness related research issues. Section 5 concludes the chapter.

Table 1.1 summarized the main QRs aspects to be covered in this chapter.

Table 1.1 Main QRs aspects

Main aspects	Description
How software QRs appeared and why?	• They appear at the requirements level, generally at the specification step of FRs and are integrated in the "Requirements Specification Document" (RSD); • The evolving technology, industry experience, costly applications in time and resources, retuned back application and rarely respect time deadline, dissatisfied users and limitations of existing requirements engineering techniques (viewpoint & object oriented) in addressing software QRs led to the critical need to recognize and address QRs; • Easy to specify but difficult to represent and control.
Why are they critical to the software engineering community (Doerr, 2011)?	NFRs are essential for software and system development • -Architecture; • -Early quality assurance; • -Subcontracting. Neglecting NFRs can lead to • failed projects; • -bad product quality; • -increased time to market (TTM); • -high rework costs.
Where are they defined or specified?	• In RUP as supplementary specifications (Jacobson et al., 1999); • In the last section of the «Use Case» description; • In the requirements specification document and annexed as quality constraints.
What is the terminology used to specify them?	• "NFR describes a certain value (or value domain) for a QA that should be achieved in a specific project. The NFR constraints a QA by determining a value for a metric associated with the QA." (Doerr et al., 2005) ; • User needs representing the design and end user views (Felici et al., 2000) ; • Quality goals and characteristics of the software product (Trendowicz et al., 1998) and (Punter et al., 2000); • Global properties of a system, assumptions, quality constraints or goals of stakeholders (Brito et al., 2002); • QAs (Doerr et al., 2005) and (Kazman et al., 2000) ; • NFRs in SWEBOK (Abran et al., 2004); • The most popular and recognized terminology is NFR and QAs where NFRs are instantiation of QAs.
Who is interested by them?	• Architect, Maintainer, Developer, Manager, Evaluator; • Customer; • End user.
Are there any research motivations in this direction?	• ISO/IEC 9126 for software product: • Good reference but needs to be supported by practical guidelines and structured methods; • There is also a need to process to map quality concepts of the standard with the product definition phase.
What are the important QRs management methods/quality standards developed during the 2 last decades?	• FDAF method (Dai, 2005); Quality model for quality attribute (Brito et al., 2002); • MOQARE method (Herrmann et al., 2007); ATAM method (Kazman et al., 2000); • IESE-NFR method (Doerr et al., 2005); Soft Goal notation (Chung et al., 2000); • ISO SQuaRE 25030 standard ; BMM (Business Motivation Model) (BRG, 2007).

1.2 Quality requirements

This section is focused on quality requirements in software engineering with emphasis on definitions of requirements and software quality. Both software engineering-related literature positions and software quality engineering standards are presented and analyzed.

1.2.1 Quality requirements and software quality

Before delving into the details of software requirements, it is important to define a requirement versus a need. As described by Azuma in his article (Azuma, 2004):

> "Needs for a product are expectations of stakeholders for the effects of the product when it is actually operated, which means such action to the software product as development, distribution, release, installation, use and maintenance". (Azuma, 2004).

Therefore according to Azuma, needs are divided into stated needs and implied needs and should be transformed into requirements. Furthermore, the author (Azuma, 2004) clarifies the relationships between needs and requirements by defining requirements as "Requirements are the external specification of specific needs that a product is expected to satisfy" The relationship between needs and requirements is illustrated in Figure 1.1. Stakeholder's needs (stated and implied) are collected and identified, then selected and specified to be transformed into QIU requirements, functional requirements and quality requirements.

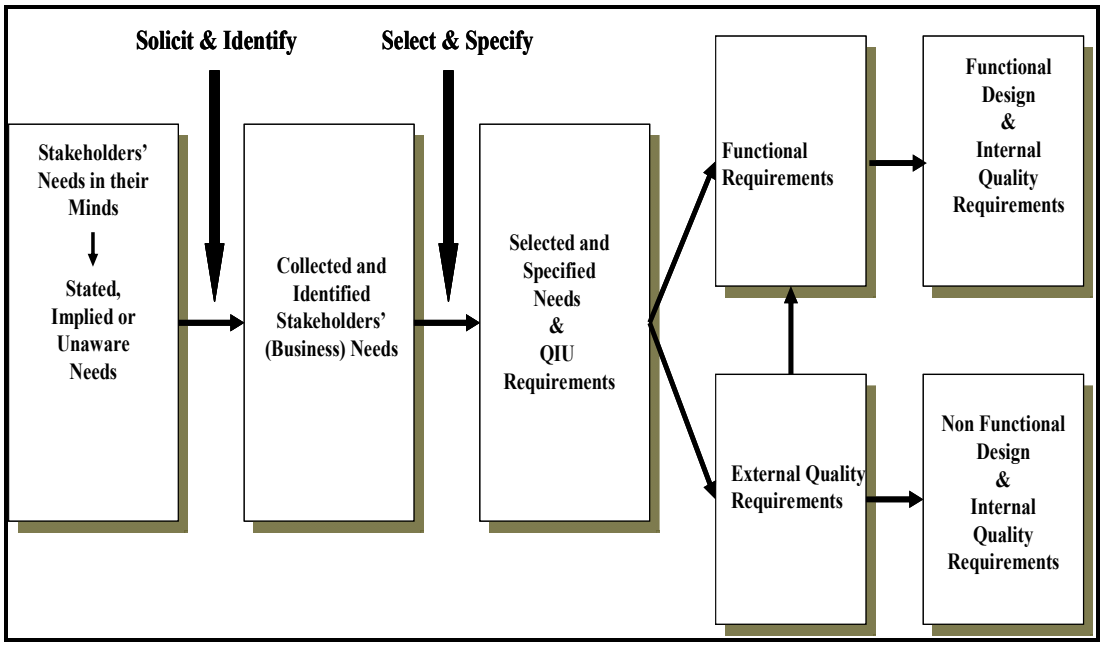


Figure 1.1 Relationships between Needs and Requirements
Extracted from Zubrow (2004)

Stakeholder needs come from many sources (Zubrow, 2004 and ISO/IEC SQuaRE 25030, 2007) (Figure 1.2). Requirements elicited from the stakeholders contribute to the definition of the three views of software quality requirements: quality in use requirements (QIU), external quality requirements (EQ) and internal quality requirements (IQ). QIU is the user's view of the quality of the software product when it is used in a specific environment and in a specific context. EQ is the totality of the characteristics of the software product from an external view. External metrics address properties visible to the users of a product (customer, manager and software engineer) such as reliability, functionality, performance and usability.

For example, reliability of the entity "operating system" can be evaluated by measuring the Mean Time To Failure (MTTF) and Rate of OCcurrence Of Failures (ROCOF). External metrics are not available until the late stages of the software development life cycle. IQ is the totality of the characteristics of the software product from an internal view. Internal metrics address properties visible only to the development team. They include size metrics (Lines of Code, number of modules) and complexity metrics (Cyclomatic complexity). The quality of the source code can be evaluated by the number of faults found by KLOC. Analyzability of

the source code can also be evaluated by the following code analysis metrics: cyclomatic number, number of statements and comment rate. Internal metrics are used to estimate external metrics at the early stages of the development process. The quality in use requirements influences the external quality, which in turn influences the internal quality requirements. The internal quality requirements are implemented through internal measures, which contribute to the specification of the external quality and quality in use of the software product.

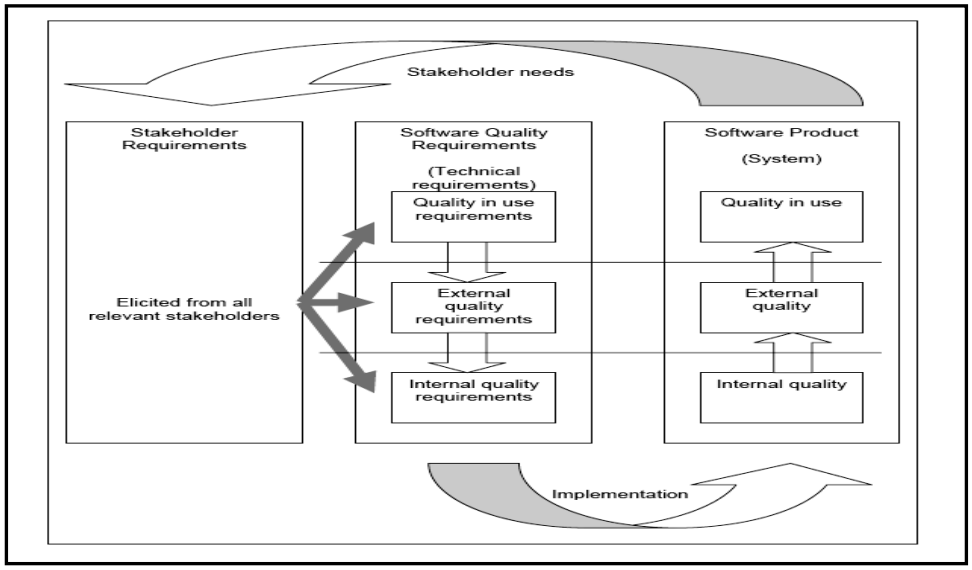


Figure 1.2 QRs life cycle model
Extracted from ISO/IEC 25030 (2007)

Moreover, the guide to SWEBOK (Software quality knowledge area (KA), chap 11) (Abran et al., 2004) describes software quality as follows:

> "What is software quality, and why it is so important that it be pervasive in the guide to SWEBOK? Over the years authors and organizations have defined the term "quality" differently. To Phil Crosby (Crosby, 1979), it was "conformance to user requirements." Watts Humphrey (Humphrey, 1989) refers to quality as "achieving excellent levels of fitness for use", while IBM coined the phrase "market-driven quality" which is based on achieving total customer satisfaction". (Abran et al., 2004).

So, according to the guide to SWEBOK, software quality refers to user requirements or levels of fitness for use or for customer satisfaction. In all these definitions the same key points are considered: requirements and stakeholder needs. Furthermore, the guide points out the definition of quality concepts and the ability of the software engineer to understand them when developing or maintaining software as it is written:

> "Thus, the software engineer has a responsibility to elicit quality requirements which may not be explicit at the outset and to discuss their importance as well as the level of difficulty in attaining them". (Abran et al., 2004).

The quality requirements should be defined and specified by the software engineer.

At the same time, the guide to SWEBOK (Abran et al., 2004) (software requirements KA, section software requirements fundamentals) defines software requirement as: "A software requirement is a property which must be exhibited by software developed or adapted to solve a particular problem".

Later in the guide, a definition of non-*functional* requirements is given as: "Non-functional requirements are the ones that act to constrain the solution. Non-functional requirements are sometimes known as constraints or quality requirements". These definitions associate non-functional requirements to quality requirements.

In summary, software quality requirements have emerged in the last decade when requirements engineering activities encountered difficulties in capturing all the fulfilled requirements (functional, performance, interface, organizational and quality). First, this difficulty was associated with non-functional requirements and supplementary requirements which were attached to functional requirements (Westerheim et al., 2005). In effect, non-functional requirements were defined as: constraints, limitations, specifications or performance (the system should run on UNIX, the system should work in real-time, the system should handle up to 500 GB of data). Later, these non-functional requirements were associated with quality requirements where more research was concentrated on their

modeling and representation (Mylopoulos et al., 1992, Dieter, 1998 and Jacobs, 1999) and on negotiation of conflicts between different requirements.

In his article (Doi, 1999), the author tries to extract quality requirements in a capturing method. The capturing method is organized around a requirements capture meeting, which is taken by videotape. Furthermore, the authors (Yuen Tak Yu and Pak-Lok Poon; 2005) present a design for the learning activities in a course on software quality practices. Their purpose is to provide opportunities for students to gain hands-on experience on exemplar software quality practices in spite of the various constraints.

On the other hand, Suryn enunciates that "Identifying quality requirements that can be elicited, formalized and further evaluated in each phase of full software product life cycle thus becomes a crucial task in the process of building a high quality software product" (Suryn et al., 2005b). This expression shows the importance of identifying quality requirements early in the software product life cycle to obtain the required software quality.

Sousa promotes separation of the concerns principle which is difficult to apply at the requirement level due to the strong relationship and interdependencies among non-functional requirements (NFRs) (Sousa et al., 2004). The basic idea is that NFRs are often scattered and tangled with the functional artifacts they affect. They describe an approach representing NFRs as concerns and compose them with the functional requirements they affect. The "Use Case" approach is used to capture and represent functional requirements (FRs) combined with the NFRs framework (Chung, 2000) to deal with NFRs concerns.

In the same research field, Cooper deals with multiple concepts to define NFRs (Cooper et al., 2005). Cooper describes an approach which integrates a semi formal UML with a set of existing formal methods into an aspect oriented framework in order to design and analyze NFRs. Cooper also considers the software architecture design as an important contribution in the reduction of development costs and the improvement of software quality. However, Cooper mentions that software architects are faced with problems of how to meet the NFRs (while designing software architecture) and argues that NFRs have to be met in order to help

designers with a rationale for decision making among competing designs. To address this problem, the author presents the Formal Design and Analysis (FDAF) Framework where NFRs are defined as reusable aspects to design and analysis and UML is extended (by stereotypes, tagged values or constraints) to support the design of these aspects. Design of aspects is performed by transforming the UML designs into formal methods and by using Chung's NFRs framework.

Otherwise, the literature shows that non-functional requirements cannot be treated alone and several authors (Paech et al., 2002 and 2003) and (Doerr et al., 2003) have argued for integrating functional (FRs), non-functional requirements (NFRs) and architectural options (AOs) early in the development process. For instance, these authors indicate that NFRs, FRs and architectural decisions must be developed in a tightly integrated approach combining elicitation, specification (of NFRs and FRs) and design architecture. Integration is supported by different kinds of experience-based artifacts such as checklists, patterns and rationale. Checklists and questionnaires are used to capture important NFRs. Architectural patterns are applied in reusing architectural options and for evaluating them against specified requirements. Traceability and rationale management are used for capturing the decision making involved in the joint specification and design of FRs, NFRs and AOs.

Paech also deals with important issues to be solved in integrating the requirement engineering process into the architectural development process (Paech et al., 2003) such as: a) use the win-win approach to identify the essential NFRs, FRs and AOs and the different views of different stakeholders; b) use goal graphs to specify NFRs and FRs and identify their dependencies and use case maps for describing AOs; c) use rationale management (goal graphs, concordance matrix and ATAM - Architectural Tradeoffs and Analysis Method) to assess how well the different AOs address a specific set of FRs and NFRs. Goal graphs are used to capture criteria (business goals) and issues (NFRs and FRs), AOs and their assessments. A concordance matrix captures assessments of the AOs against FRs and NFRs. ATAM captures criteria (quality attributes, business goals), issues (risks), options (architectural views) and assessments (utility tree). CBAM (Cost Benefit Analysis Method) is

used to refine the ATAM results with cost benefits (criteria, options); and d) use architectural styles to capture and use experience on typical AOs.

In addition, in his position paper, Mylopoulos puts emphasis on the increasing use of the "goal concept" in requirement engineering methods and techniques (Mylopoulos, 1998) considering goals as an important construct in different areas of requirements engineering as: a) requirement acquisition and specification: here goals are used as the main guiding concept in requirement specification; b) clarifying requirements: the goal oriented approach would allow the requirements to be refined and clarified through an incremental process; c) requirements conflicts: goals are a useful way to address conflicts among NFRs where difficult tradeoffs have to be made such as costs, performance, flexibility and usability; d) driving design: goals are an important driving force of requirements to design. In fact, the NFRs framework (Chung, 2000) uses NFRs as goals to guide the design process.

More definitions on quality requirement have been suggested in the engineering community. Authors like Pfleeger, van Vliet and Lauesen (Pfleeger, 2001; van Vliet, 2002 and Lauesen, 2001) highlight, in their research field, the importance of dealing with non functional requirements at an early stage.

Lauesen in his book entitled "Software requirements Styles and Techniques" defines quality requirements as "*Quality requirements specify how well the system must perform its functions. How fast it respond? How easy must it be to use? How secure does it have to be against attacks? How easy should it to be maintained?*" The McCall and Matsumoto quality model (McCall, 1977) (Operation, revision and transition) is used with a quality grid to find the important quality factors. The author also highlights the importance of QRs in the requirements specification step and confirms that they occupy little space because they are difficult to identify and verify. In addition, in the described case studies, QRs are presented as: a) quality properties enumerated as quality attributes related to the described system; b) or categorized as one of the goal levels (domain level, high-level and product level requirements) which are quality level requirements.

Lauesen describes several requirements elicitation techniques. For instance, Goal-Domain-Tracing is described as a checking technique which establishes a relation between business goals and domain issues (tasks or quality factors) and addresses these two points: a) which quality factors and tasks ensure that the business goal can be met? And b) what is the purpose of each task and quality factor in terms of business goals? The Goal-Task-Description is another technique used to show relationships between goals and tasks in order to identify the critical tasks. Goal-Task-Description is also used to progress from stating business goals to formulating requirements (functional and quality requirements). To formulate quality requirements (for instance, usability), the method used is : a) identify usability issues, business goals, concerns, user profiles and critical tasks; b) select requirements styles to cover the issues; c) select metrics and target values.

Pfleeger describes in her book functional and non-functional requirements and explores their characteristics and methods to define and specify them. Pfleeger defines non-functional requirement as: "*A requirement or constraint describing a restriction on the system that limits our choices for constructing a solution to the problem*". Requirements are written in a Requirements Statement Language (RSL). The author highlights how to express NFRs as descriptions of the path through the R-nets as mentioned in "*We can think of the non functional requirements as descriptions of constraints placed on the flow along various paths*". NFRs are specified by making the R-nets with validation points ("*A validation point is a place in the diagram used to denote the beginning or the end of a measurement*").
Pfleeger presents different specification techniques of requirements (ranking from static: data flow diagram to time related dependencies and oriented object) which do not mention anywhere how to identify NFRs. An exception is made for the technique SREM ("Software Requirements Engineering Methodology") which views the system as a finite state machine where the statements are analyzed by a Requirements Engineering Validation System (REVS). As enunciated by the author, "*RSL describes the flow of processing in terms of what events initiate which processes. These flows are represented as networks".* These networks or R-nets specify the transformation of a particular state and a single input into a new state

with multiple output messages. RSL allows for a complete specification view of FRs and NFRs requirements associated to elements and processing steps.

Hans Van Vliet in his book defines four types of non-functional requirements according to IEEE framework (Std 830-1993: Recommended Practice for Software Requirements Specifications): external interface requirements, performance requirements, design constraints and software system attributes. Performance requirements and software quality attributes are known as quality requirements. The author highlights the importance of these requirements and difficulty to specify and verify them. Van Vliet also emphasizes the fact that these requirements should be expressed in objective and measurable terms.

Van Vliet presents a list of techniques for capturing and formulating requirements. For example, task analysis is a technique used to obtain a hierarchy of tasks and subtasks to be carried out by people working in the domain. Scenario-based analysis is a method which analyzes, generates and validates scenarios in a systematic way. Entity-Relationship Modeling is a requirements specification technique which models the data aspect and the finite State machines are used to model the functional aspect.

Other authors are working in the same research field to address organizational requirements in conjunction with quality requirements (Firesmith et al., 2004). In their research study entitled *"Requirements Elicitation and Analysis Processes for Safety and Security Requirements*", Firesmith described the problems encountered when requirements engineering practices are missed or not well defined early in the development life cycle and mentioned also that organizational mechanisms facilitate the promotion of quality requirements in the software development process. The search focus was oriented specifically on "safety and security engineering" and aims to identify the potential conflicts of quality concepts (time consuming) with organizational mechanisms (time to market and cost considerations) and to support quality engineering by organizational techniques. An elicitation process for security requirements was developed and supported by prototype tools (Mead, 2004 and Firesmith, 2003 and 2005).

On the other hand, Bevan considers software product quality by achieving "quality in use" and adopting a user centered design process to meet user needs for quality (Bevan, 1999 and Bevan et al., 1997). A "quality in use" is defined as a way to incorporate human factors into the software engineering life cycle. In other terms, by defining "quality in use", a link is provided between the human factors approach to usability and user centered design.

Table 1.2 summarizes relevant software QRs definitions suggested by different authors.

Table 1.2 Definitions of software QRs by authors

QRs authors	Definitions
Guide to SWEBOK (Abran et al., 2001)	NFRs "Non-functional requirements" are constraints or quality requirements"
RUP (Jacobson et al., 1999)	Supplementary requirements attached to functional requirements
(Sousa et al., 2004)	NFRs as concerns according to the "Separation of concerns" principle
(Paech et al., 2002 and 2003)	NFRs as instantiation of QAs which should be integrated with FRs and architectural options
(Cooper and al., 2004)	NFRs as aspects represented in an extended UML
(Mylopulos , 1998) and (Chung et al., 2000)	"Goal concept" " Soft goal concept"
(Pfleeger, 2001)	"A requirement or constraint describing a restriction on the system that limits our choices for constructing a solution to the problem".
(Hans Van Vliet, 2002)	Four types of NFRs according to IEEE framework (Std 830-1993: Recommended Practice for Software Requirements Specifications): external interface requirements, performance requirements, design constraints and software system attributes.
(Lauesen, 2001)	"Quality requirements specify how well the system must perform its functions. How fast it respond? How easy must it be to use? How secure does it have to be against attacks? How easy should it to be maintained?"
(Firesmith and al., 2004)	Safety and Security engineering
(Bevan and al., 1997) and (Bevan, 1999)	Usability and quality in use

In conclusion, one can say that quality requirements have been addressed by different authors and most of them put emphasis on their importance in determining the software product quality. These authors also turn the reader's attention to the difficulty of identifying quality requirements and to the need for developing more methods and approaches to deal with them.

1.3 Software quality engineering standards

One of the first predecessors of today's quality models is the quality model presented by Jim McCall (McCall et al., 1977) (also known as the General Electric's Model of 1977). This model originates from the US military and is primarily aimed toward the system developers and the system development process. McCall attempts to establish a link between users and developers by defining a number of software quality factors that reflect both the users' views and the developers' priorities. The McCall quality model has three major perspectives for defining and identifying the quality of a software product: product revision (ability to undergo changes), product transition (adaptability to new environments) and product operations (operational characteristics).

The model details the three major perspectives in a hierarchy of a) factors (to specify) which describe the external view of the software as viewed by the users, b) criteria (to build) which describe the internal view of the software as seen by the developer and c) metrics (to control) which are defined and used to provide a scale and method of measurement.

The second of the basic predecessors of today's quality models is the quality model presented by Barry W. Boehm (Boehm et al., 1978). Boehm's model attempts to qualitatively define software quality by a given set of attributes and metrics. Boehm's model presents a hierarchical quality model structured around high-level characteristics, intermediate level characteristics, and primitive characteristics.

The high-level characteristics represent basic high-level requirements of actual use to which an evaluation of software quality could be put – the general utility of software.
The intermediate level characteristic represents Boehm's 7 quality factors that together represent the qualities expected from a software system.

The lowest level structure of the characteristics hierarchy in Boehm's model is the primitive characteristics metrics hierarchy. The primitive characteristics provide the foundation for defining quality metrics.

A more recent model is the quality model presented by R. Geoff Dromey (Dromey, 1995). His idea is as quoted “quality evaluation differs for each product and that a more dynamic idea for modeling the process is needed to be wide enough to apply for different systems”. Dromey’s quality model is based on the relationship between quality attributes and sub-attributes, as well as the connexion between the product properties and the software quality attributes.

The standard ISO/IEC 9126 (ISO/IEC 9126, 2004) developed by ISO/IEC JTC1 SC7 (Subcommittee SC7 - Software and Systems Engineering of International Organization for Standardization) is divided into four parts:

1. ISO/IEC 9126-1: Information technology - Software quality characteristics and metrics - Part 1: Quality model.

This part provides the recommended quality model containing important quality characteristics for the final product. Quality sub characteristics and attributes refine the quality model and can be internal or external quality attributes.

2. ISO/IEC 9126-2: Information technology - Software quality characteristics and metrics - Part 2: External metrics (Figure 1.3).

This part provides external quality metrics for measuring software quality characteristics applicable to an executable software product during testing or operating at a later stage of development and after entering the operation process.

3. ISO/IEC 9126-3: Information technology - Software quality characteristics and metrics - Part 3: Internal metrics (Figure 1.3).

This part provides internal quality metrics for measuring software quality characteristics applicable to a non-executable software product during designing and coding at an early stage of the development process.

4. ISO/IEC 9126-4: Information technology - Software quality characteristics and metrics - Part 4: Quality in use metrics (Figure 1.4)

This part provides quality in use metrics for measuring software quality characteristics applicable to an executable software product after entering the operation process.

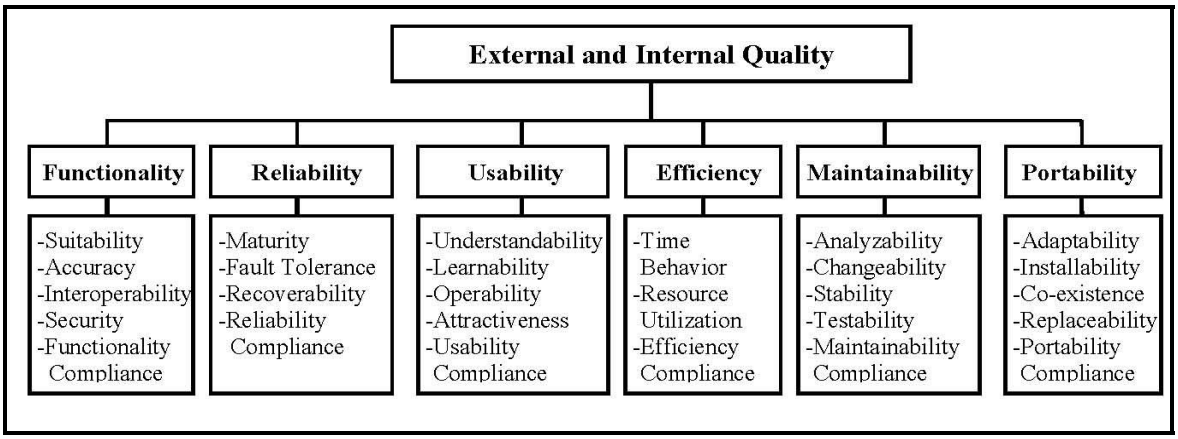


Figure 1.3 ISO/IEC 9126 Quality Model - External and Internal Quality
Extracted from Suryn et al., (2005b)

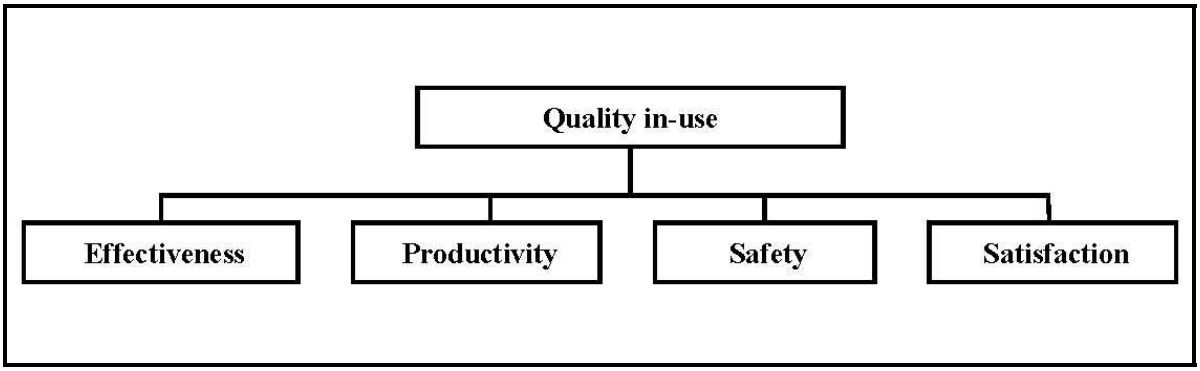


Figure 1.4 ISO/IEC 9126 Quality Model - Quality in Use
Extracted from Suryn et al., (2005b)

Azuma presents in his article (Azuma, 2004) the categorization of software quality requirements according to ISO/IEC JTC1/SC7 (External Quality Requirements, Internal Quality Requirements, and Quality-In-Use Requirements) as follows:

> "External Quality Requirements specify the required level of quality from the external view. They include requirements derived from user quality needs, including Quality-In-Use requirements". "Internal quality requirements are used to specify properties of interim products, including static and dynamic models, other documents and source code". (Azuma, 2004).

Moreover, according to Suryn, the extraction of software quality requirements begins with identification of stakeholder requirements and continues through decomposition until all corresponding categories of quality requirements are identified (Quality in use, external quality, internal quality and operational quality) (Suryn, 2003). The quality requirements decomposition model is static and gives no insight on how to extract and decompose quality requirements (Figure 1.5). Hence, the process of defining and controlling quality requirements has been proposed to state important questions to be asked about the way to define and control quality requirements (Figure 1.6).

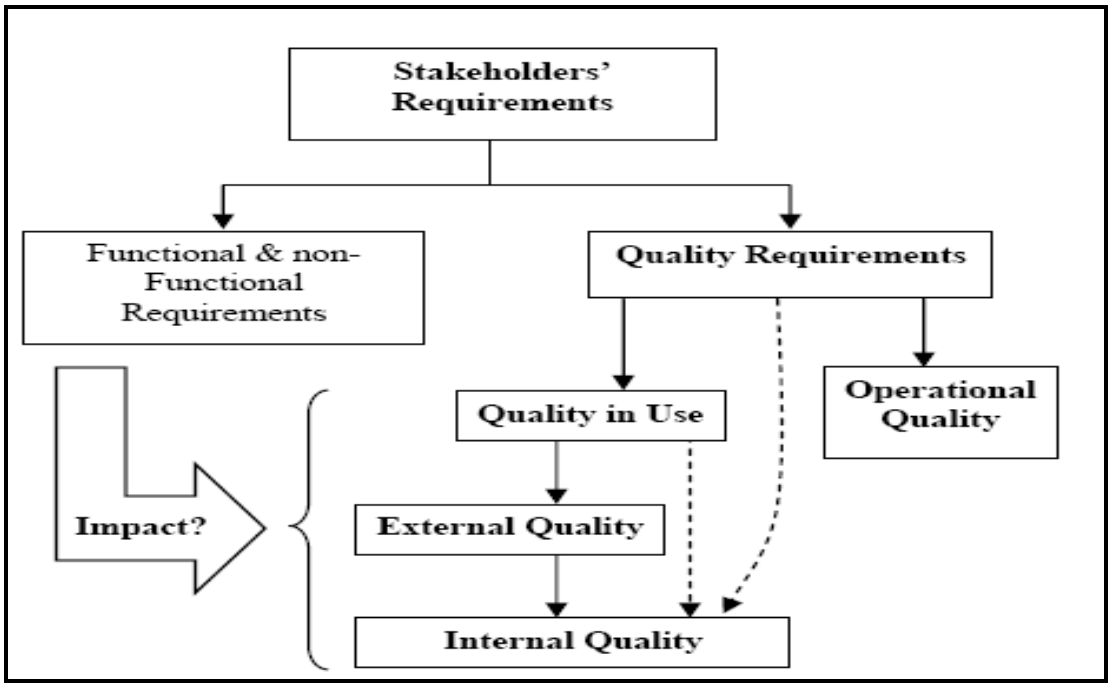


Figure 1.5 Quality requirements decomposition model
Extracted from Suryn (2003)

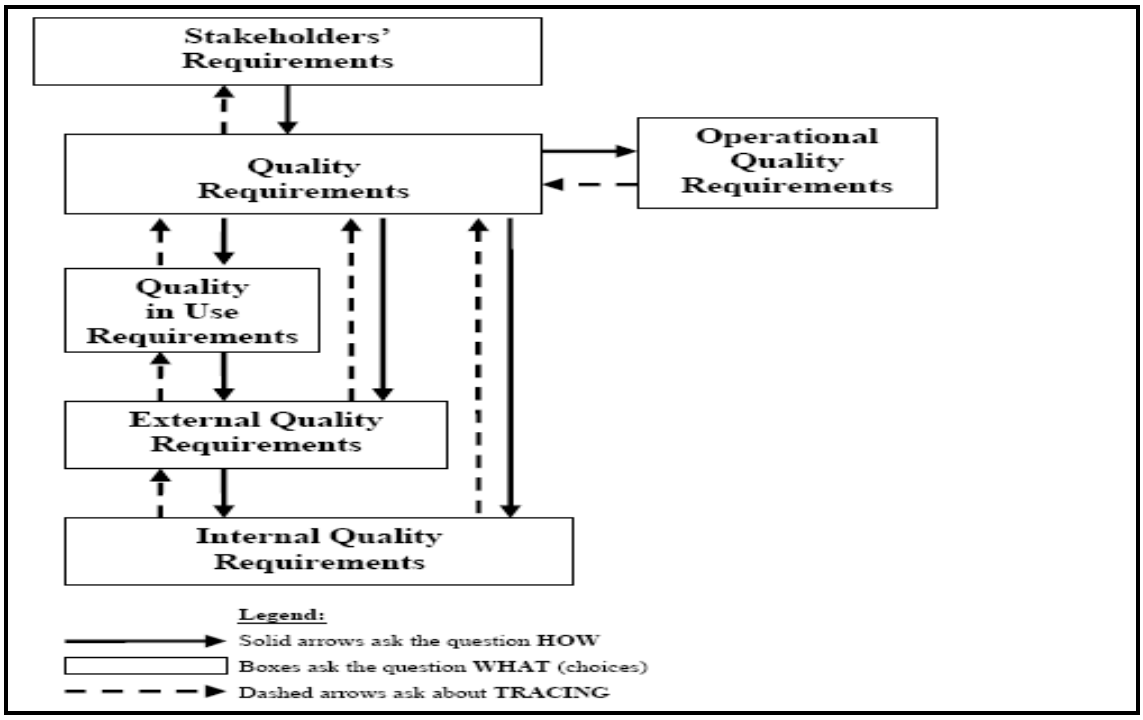


Figure 1.6 Process of defining and controlling quality requirements
Extracted from Suryn (2003)

Operational quality (OP) shown in Figures 1.5 and 1.6 is described by TL 9000 standards: TL 9000 Quality System Requirements (TL900, 2001a) and TL 9000 Quality System Measurements (TL900, 2001b). These standards are developed by QUEST Forum (in 1999-2000 and Published in 2001) for the set of initial requirements for operational quality as well as for reporting on implemented quality once the software product has been developed and deployed in the field (Suryn et al., 2004a.)

TL 9000 (part 2) identifies four categories of requirements and/or measurements applicable to software products:

1. Common measurements – referring to the number of problems reported, response time, overdue problem responsiveness and on-time delivery;
2. Hardware and software measurements – referring to system outage;
3. Software measurements – referring to software installation and maintenance;
4. Service measurement – referring to service quality.

Figure 1.7 illustrates the TL 9000 model structured in layers:

1. International Standard - ISO 9001;
2. Common TL 9000 Requirements;
3. Hardware, Software and Services Specific Quality System Requirements;
4. Common TL 9000 Metrics;
5. Hardware, Software and Services Specific Quality System Metrics.

Figure 1.7 Quest FORUM TL9000 Model
Extracted from TL 9000 (2001)

The TL 9000 standard series was combined with ISO/IEC 9126 to create CQL (Consolidated quality life cycle) model (Figure 1.8), which serves as the basis (backbone) for the process of defining quality requirements, their measurement and evaluation. CQL model was proposed by Suryn and Abran (Suryn et al., 2004a.) where they describe the applicability of a CQL model in each phase during the development process. The Discovery and Requirements Analysis phases are briefly presented hereafter.

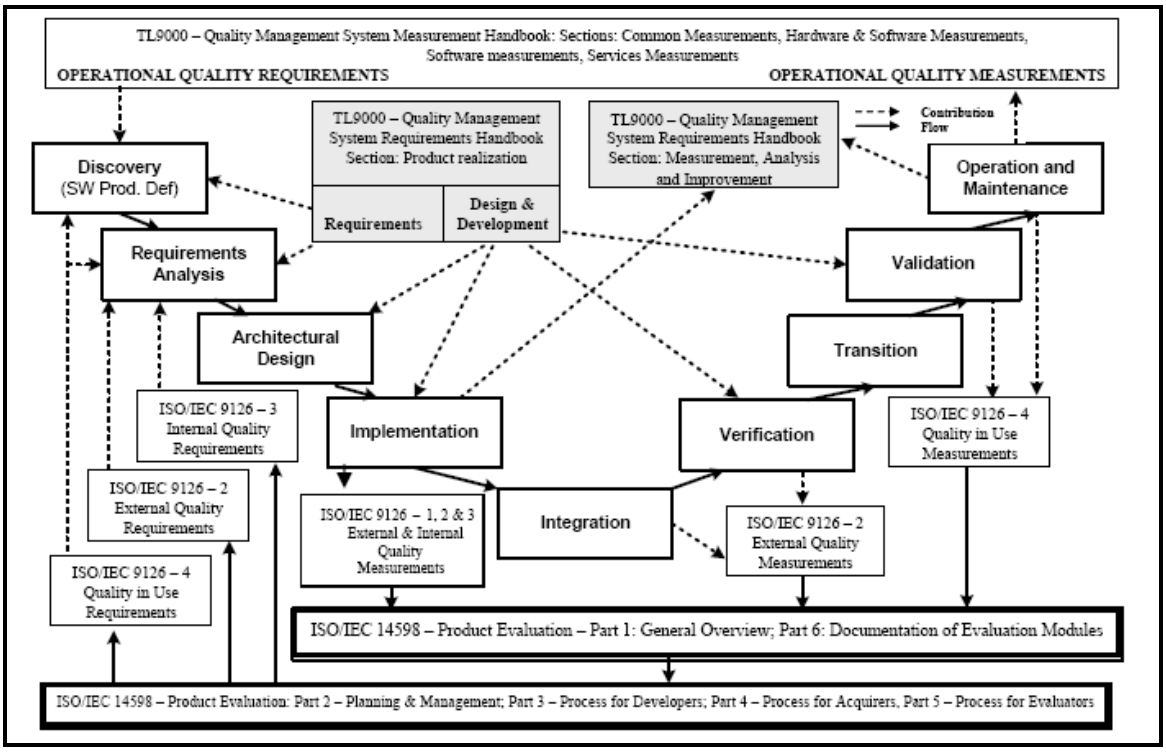


Figure 1.8 Suryn-Abran CQL model version 1.1
Extracted from Suryn et al., (2005)

Discovery Phase: The definition of quality requirements is undertaken in the Discovery Phase. Three sets of requirements have to be identified and defined:

1. Functional and non-functional requirements of the product;
2. Operational quality requirements;
3. Quality in Use requirements.

In this phase "quality in use and operational quality" characteristics are analyzed and applicable measures are defined. Target values are then assigned for each. Standards to be applied to complete this task are ISO/IEC 9126 – Part 4: Quality in Use Metrics and TL 9000 – Quality Management System Measurement Handbook (part 2).

Requirements Analysis Phase: In this phase external and internal quality attributes of the software product are defined. The ISO standards applied in this phase are:

1. ISO/IEC 9126 – Part 2: External Quality Metrics
2. ISO/IEC 9126 – Part 3: Internal Quality Metrics

The CQL model was improved after several steps resulting from detailed analysis and verification. The authors (Suryn and al, 2005b.) present a research model analysis and propose enhancements (normative support) for each phase of CQL model (Figure 1.8).

1.3.1 Software quality Requirements and ISO/IEC SQuaRE standard

In their research study (Suryn and Abran, 2003), the authors addressed the need to integrate process and product standards in the development process through their quality engineering approach. They highlighted the absence of ISO standards used in the product definition phase (Figure 1.9) and the mapping mechanisms between these standards and all phases of the life cycle of a software product. The ISO/IEC 15288 – *System life cycle processes* (ISO/IEC 15288, 2002) which identifies the generic phases of the development process was integrated into these standards in order to define the mapping between these standards and the software product life cycle phases.

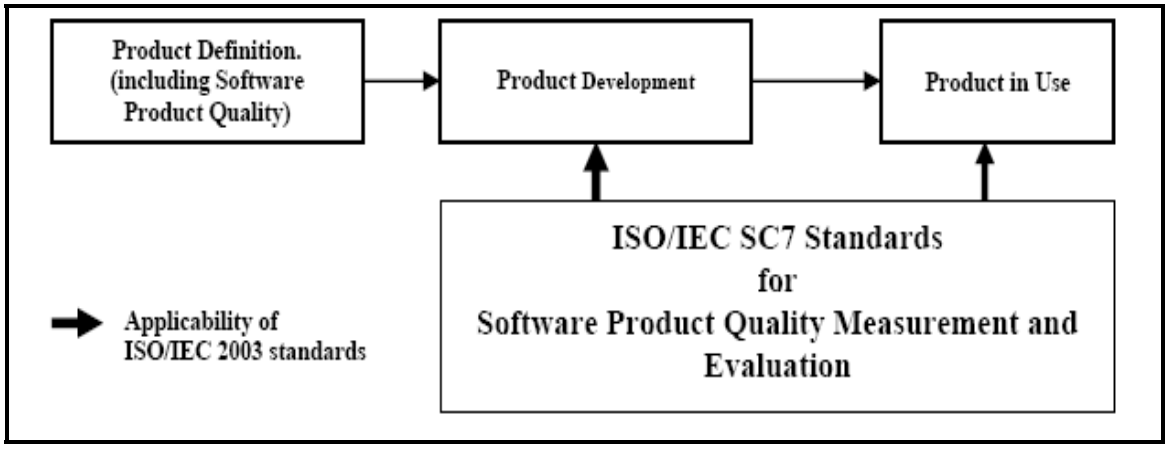


Figure 1.9 High-level mapping of ISO/IEC SC7 software product quality Standards and a software life cycle
Extracted from Suryn et al., (2003)

The first generation of software quality engineering standards developed by the ISO SC7 (ISO/IEC 9126 – *Software Engineering – Product quality* and ISO/IEC 14598 - *Evaluation of software products)* presents some limitations (Figure 1.10) as mentioned:

"While it provides generic linkages between the high-level concepts of the ISO 9126 quality instruments (i.e. characteristics, sub characteristics and measures), it is not yet specified in the format of specific prescriptive quality engineering practices. In particular, the current versions of these ISO/IEC standards do not provide a clear mapping between the quality engineering instruments already developed and the various phases of the product development life cycle". (Suryn and Abran, 2003).

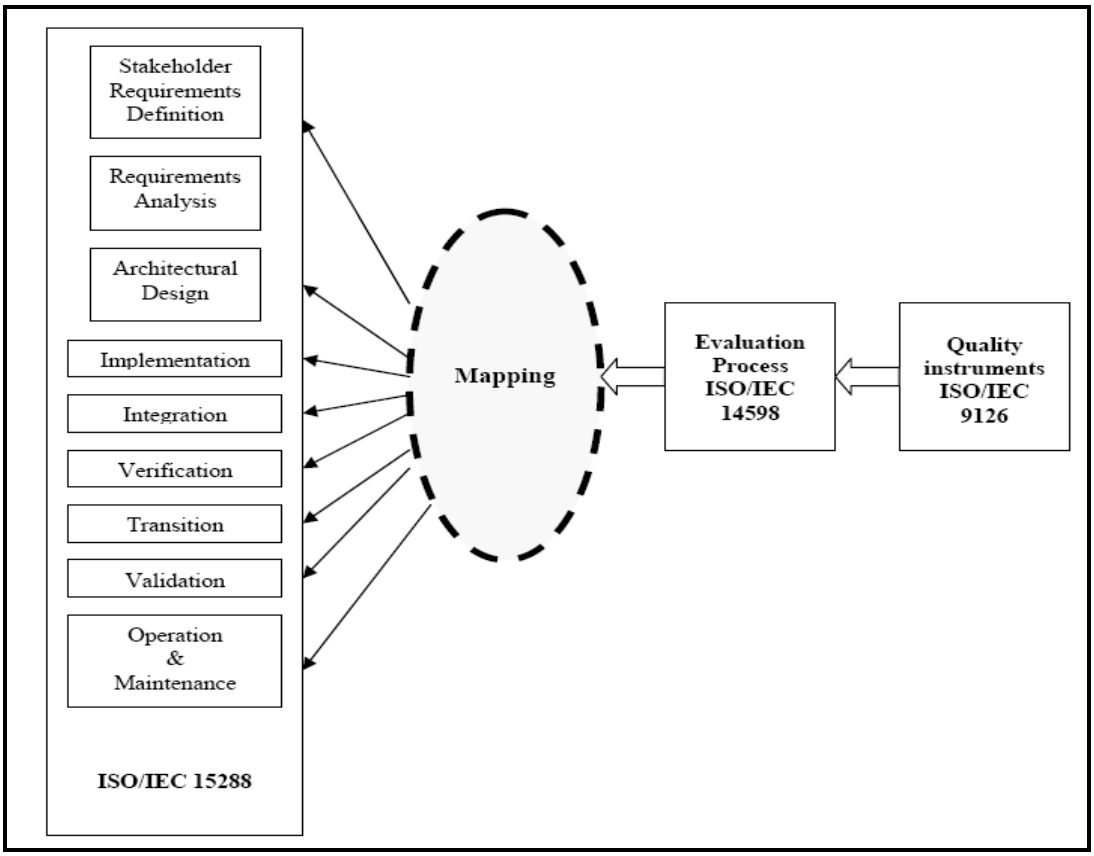


Figure 1.10 Mapping between ISO/IEC 15288, ISO/IEC 9126 and ISO/IEC 14598
Extracted from Suryn et al., (2003)

In fact, the standards provide the static quality concepts but do not support the mapping between quality concepts and the software life cycle phases. Based on these remarks, the authors present the relevant improvements proposed by the ISO/IEC SC7 WG6 experts to build the new standard for software quality requirements specifications ISO/IEC SQuaRE 25000 – Software Product Quality Requirements and Evaluation.

1.3.2 Standard ISO/IEC SQuaRE 25030 - Software Product Quality Requirements

ISO/IEC SQuaRE: 25000: The Software Product Quality Requirements and Evaluation standard is a set of international standards and technical reports on software product quality. SQuaRE consists of five divisions: quality management, quality requirements, quality evaluation, quality models and quality metrics. This standard includes: definitions of terms, reference models and a general guide, requirements and recommendations, and individual guides for the use of the series.

ISO/IEC SQuaRE: 25030 is described by (Azuma, 2001):

> "Quality Requirements is a "SQuaRE" standard that enables software product quality requirement to be specified, tracked, validated and managed with evaluation from different perspectives by those associated with acquisition, requirements analysis, development, use, evaluation, support, maintenance, quality assurance and audit of software. It provides a guide to use the model and metrics for requirement definition". (Azuma, 2001).

Azuma also indicates that the application of SQuaRE 25030 standard allows one to:

- *"*Validate the completeness of a requirements definition;
- Identify software requirements from a view of quality;
- Identify software design objectives;
- Identify software testing objectives;
- Identify acceptance criteria for a completed software product*"*. (Azuma, 2001).

The quality requirements components of the standard are presented in Figure 1.11.

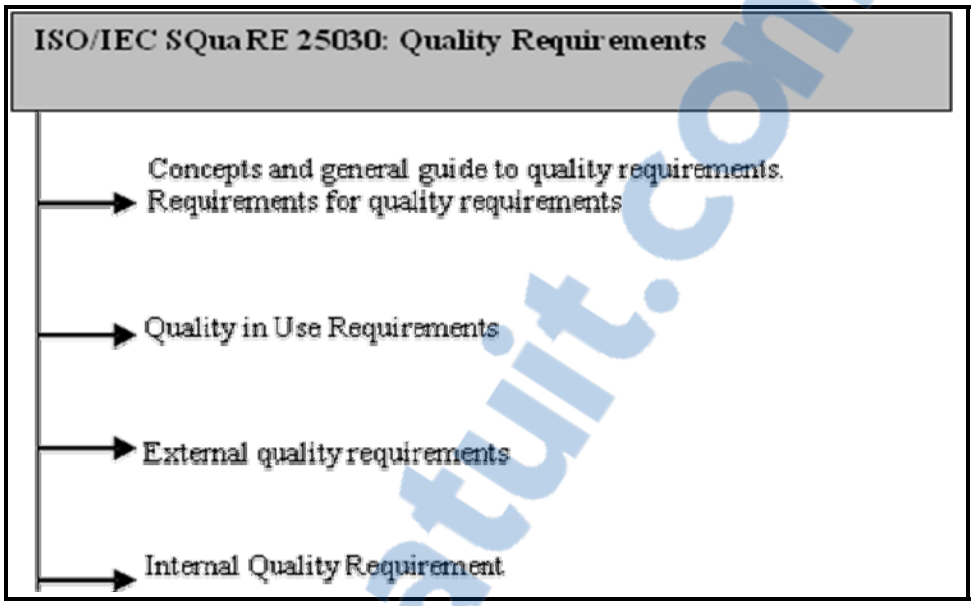


Figure 1.11 ISO/IEC SQuaRE 25030 Quality Requirement Division
Extracted from ISO/IEC 25030 (2007)

The steps of the standard (ISO/IEC 25030, 2007) are listed as follows (Figure 1.12):

- General assumptions;
- System considerations;
- Stakeholder's considerations;
- Quality model considerations;
- V&V considerations.

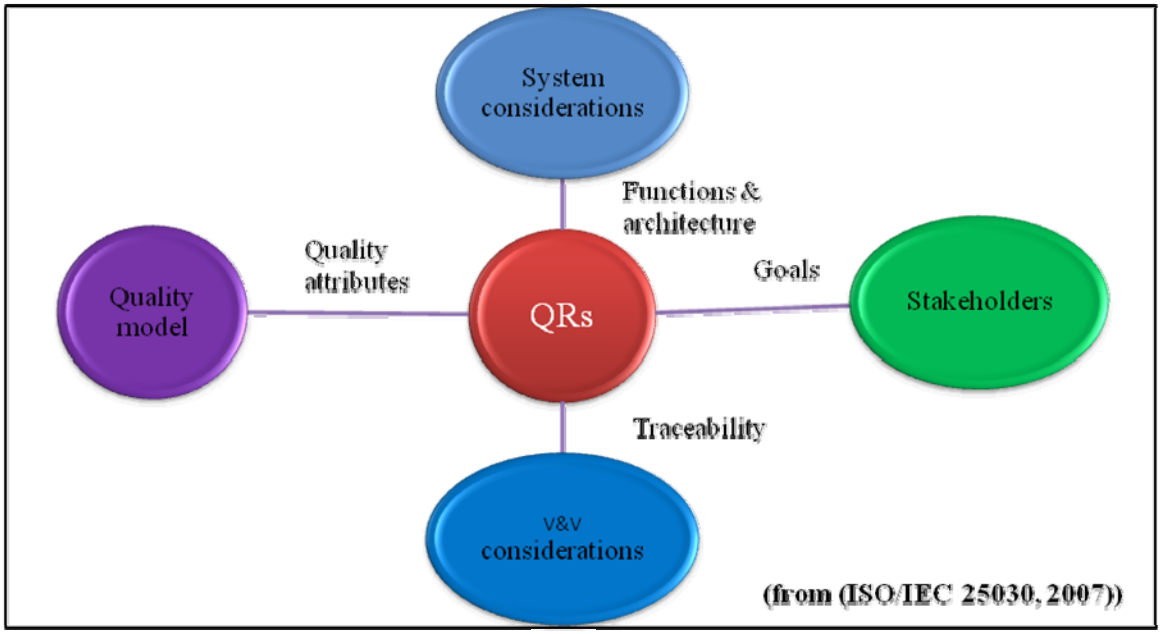


Figure 1.12 Steps of the standard

System considerations are represented in Figure 1.13

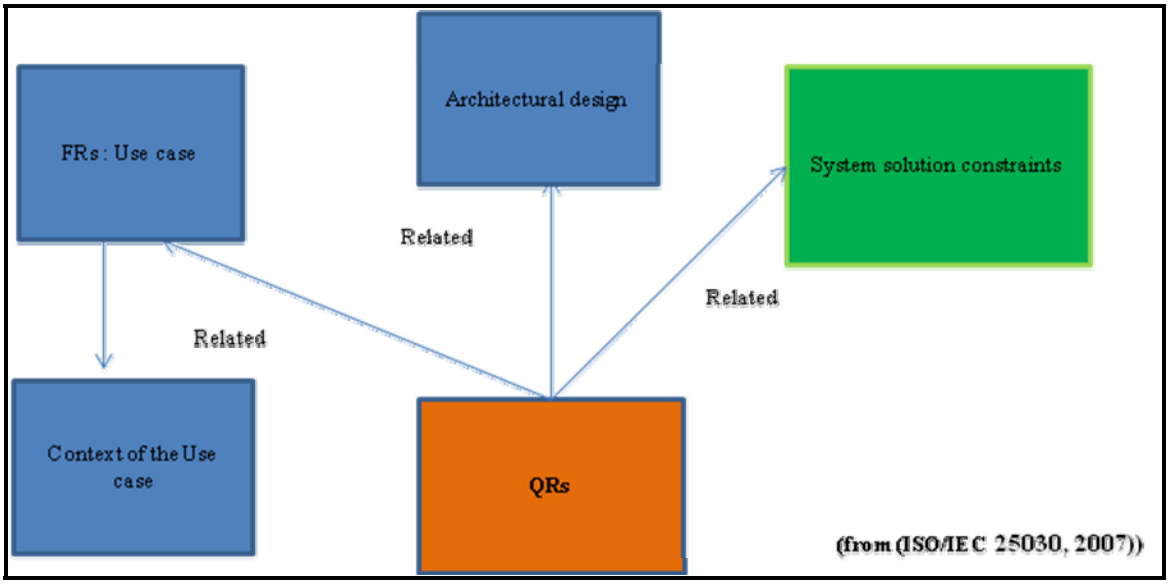


Figure 1.13 System considerations

Stakeholder's considerations are represented in Figure 1.14.

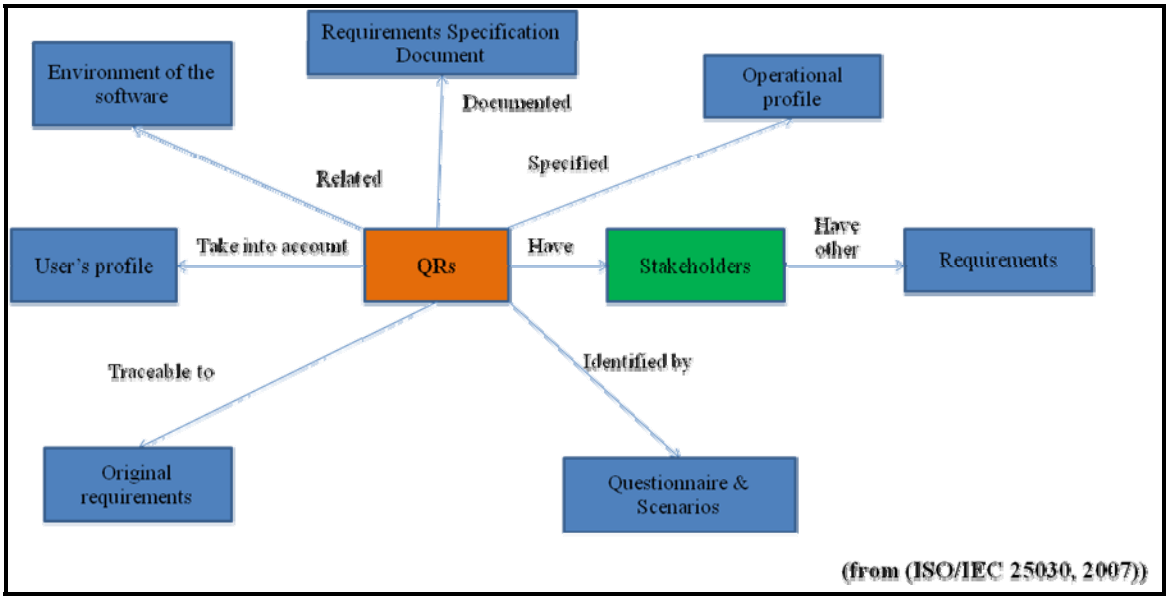


Figure 1.14 Stakeholders considerations

Quality model considerations are represented in Figure 1.15

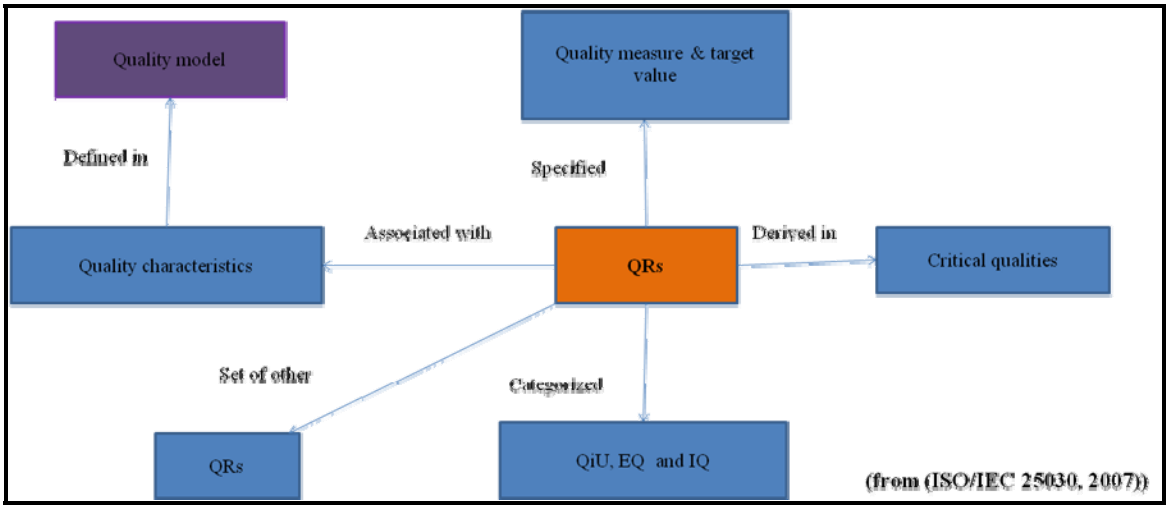


Figure 1.15 Quality model considerations

Validation and verification (V&V) considerations are represented in Figure 1.16

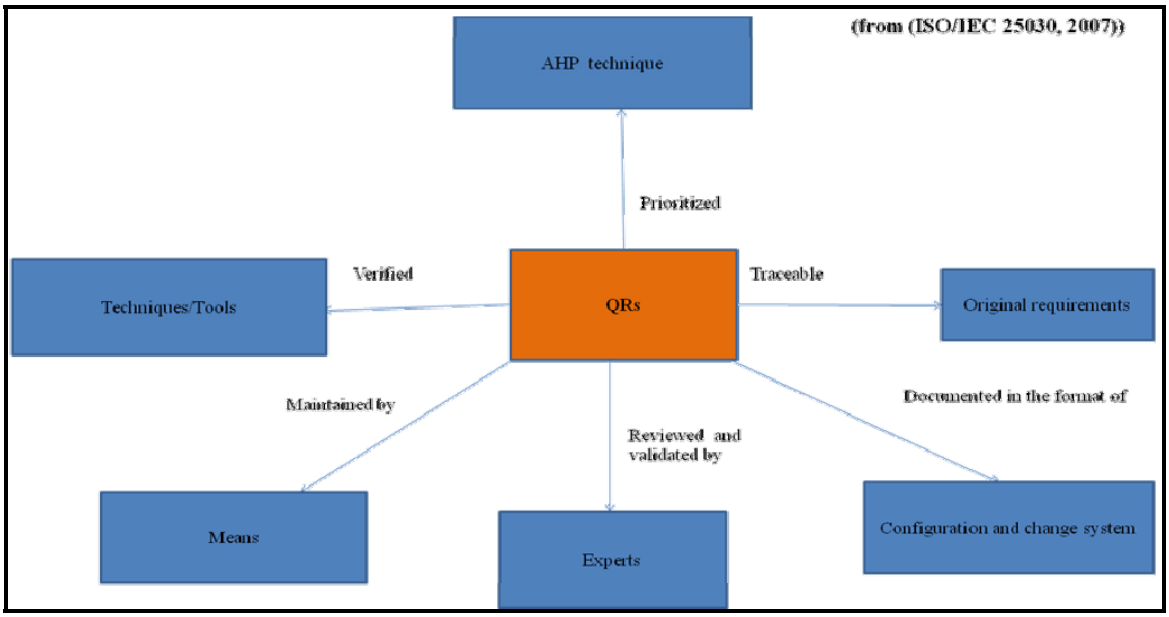


Figure 1.16 V&V considerations

As SQuaRE complies with ISO/IEC 15288 System Life Cycle Processes, Azuma proposes a contribution from the guide ISO/IEC 25030: Quality requirements in the phases: "Stakeholder requirements definition" process and "Requirements analysis" process (Figure 1.17).

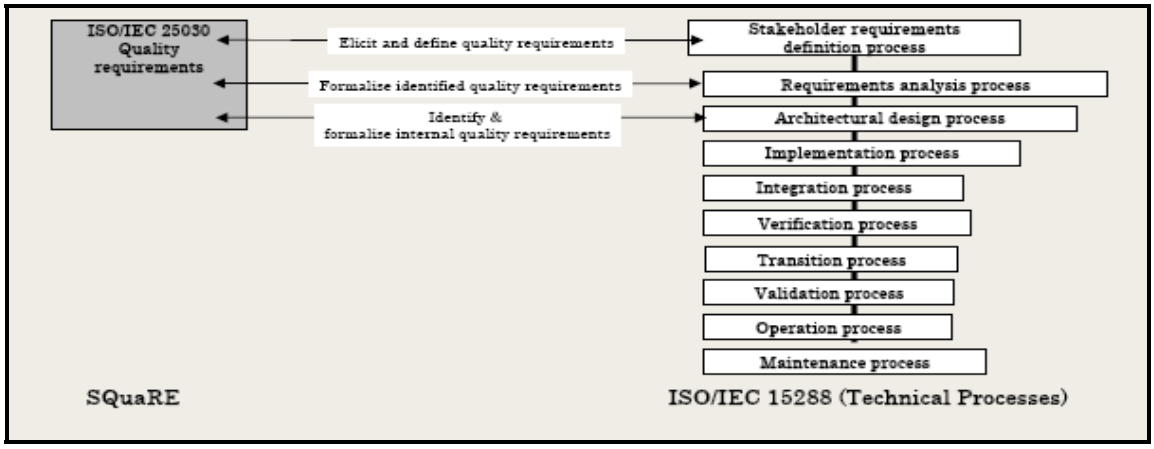


Figure 1.17 ISO/IEC 15288 System Life Cycle Processes to appear in 25030
Extracted from Zubrow (2004)

The proposed contribution from the guide ISO/IEC 25030 is described by the following activities:

- Elicitation and definition of quality requirements as input to “Stakeholder requirements definition” process;
- Formalization of identified requirements as input to “Requirements analysis” process;
- Identification and formalization of internal quality requirements.

Software quality engineering standards have proven their applicability in different fields of application such as facilitation of communication between users through a standard language (ISO/IEC 9126, 2004) and (ISO/IEC 14598, 1999). However, they need to be supported by techniques and practical guidelines to identify and model software QRs. As a solution, there is a possibility to combine quality standards with QRs management methods. The next section will present some QRs management methods which address elicitation and definition of quality requirements.

1.4 Quality requirements management methods

Over the past two decades, research on software quality and quality requirements (QRs) has resulted in several software QRs management methods. These methods are classified into four categories developed at two subsequent levels (requirement and architectural). This classification was based on the main drivers contributing to identify and specify quality attributes and are: business goals, aspect and goal concepts. The methods are:

a) **Business goal oriented methods** which use business goals as main drivers in the software quality process:
   a. Space-Ufo: uses business issues to identify the quality needs of the stakeholders (users, customers and managers) (requirements level);
   b. MOQARE (Misuse Oriented QuAlity Requirements Engineering): uses business goals and the misuse concept to describe quality attributes (requirements level);
   c. ATAM (Architecture Tradeoff Analysis Method): uses business goals and scenarios to describe quality attributes (architectural level).

b) **Aspect oriented methods** are based on the aspect concept of the "Aspect oriented paradigm":
   a. FDAF (Formal Design and Analysis Framework): uses the aspect concept and formal methods to design and analyse NFRs (architectural level);
   b. Quality model for quality attributes: uses the aspect concept to specify quality attributes (requirements level).

c) **Goal oriented methods** which are based on the goal concept to specify, refine and analyze conflicts:
   a. IESE NFR **(I**nstitute for **E**xperimental **S**oftware **E**ngineering for **NFR)** deals with quality attributes of embedded systems (requirement and architectural levels);
   b. Soft goal notation : uses goals as a driving force to elicit and refine NFRS and to guide the design process (requirements and architectural levels);

c. Prometheus **(**Probabilistic Method for early evaluation of NFRs**)**: combines goal concepts to operationalize quality goals via the Goal Measurement template (requirements level).

d) Other QRs management methods:

a. Quality models in software packages (requirements level);
b. Quality specification strategies for embedded systems (requirements level);
c. SHEL (**S**oftware and **H**ardwar**E** and **L**ive ware) methodology which deals with the integration of different types of requirements (functional, cognitive and quality) (requirements and architectural levels);

Each method will be described according to its process and model, analyzed and discussed by establishing strengths and weaknesses. The analysis and discussion of strengths and weaknesses are based on the existing literature on the QRs methods and on Djouab's analysis. A conclusion ends this section with important observations arising from the studied quality methods.

1.4.1 SPACE-UFO[1] Project

1.4.1.1 Description of the method

The approach presented by Punter in (Punter et al., 1997), (Veenendaal, 1997) and (Space-Ufo, 1998) deals with the fit between the software product characteristics and the user's need for that product (explicit and implicit). The authors present the SPACE-UFO project and describe the method for IT product quality requirements specifications and evaluation. This method is focused on user needs and is used as a "quality target" for both IT the product

[1] SPACE-UFO project is part of the SPACE-**S**oftware **P**roduct **A**dvanced **C**ertification and **E**valuation --**U**ser **FO**cus- is a new CEC ESPRIT project that will provide an enhanced user-oriented method for IT product quality requirements specification.

evaluation process and the IT development process. Requirements addressed by this method are quality needs and quality characteristics of the software product. The quality model used by this method is the standard ISO/IEC 9126. The tools/techniques supporting this method are: questionnaires, scenarios and interviews.

1.4.1.2 Activities of this method

The authors describe the reference model of this method (Figure 1.18). The main objective of this methodology is to specify quality requirements for the software product and to evaluate the quality of this software product. The basic idea is to use a first transformation process to elaborate a quality profile (based on ISO/IEC 9126 model) from the descriptions of the business process, the needs of the user/customer and the software product itself. A second transformation process is used to produce a quality specification (describing quality characteristics of the software product being developed which serves as input to the development process) and an evaluation plan (describing techniques and tools to be used to evaluate the software product).

Furthermore, Punter et al point out the importance of building a quality profile of the software product which is defined as a list of ISO/IEC 9126 prioritized quality characteristics and sub characteristics and a number of requirements associated with these quality characteristics. The main phases of building the quality profile are as follows:

- **Identification of quality needs:** quality needs of the stakeholders (users, customers and managers) are related to business issues or companies. The user's quality needs for a software product are defined in accordance with the influence a software product has on:
  - Business system and characteristics of that business system;
  - User tasks.
- **Specification of quality characteristics**: quality characteristics have to be specified and quantified in a consistent and complete manner. It is important to find a good definition for each sub characteristic and to link that characteristic to the associated metric.

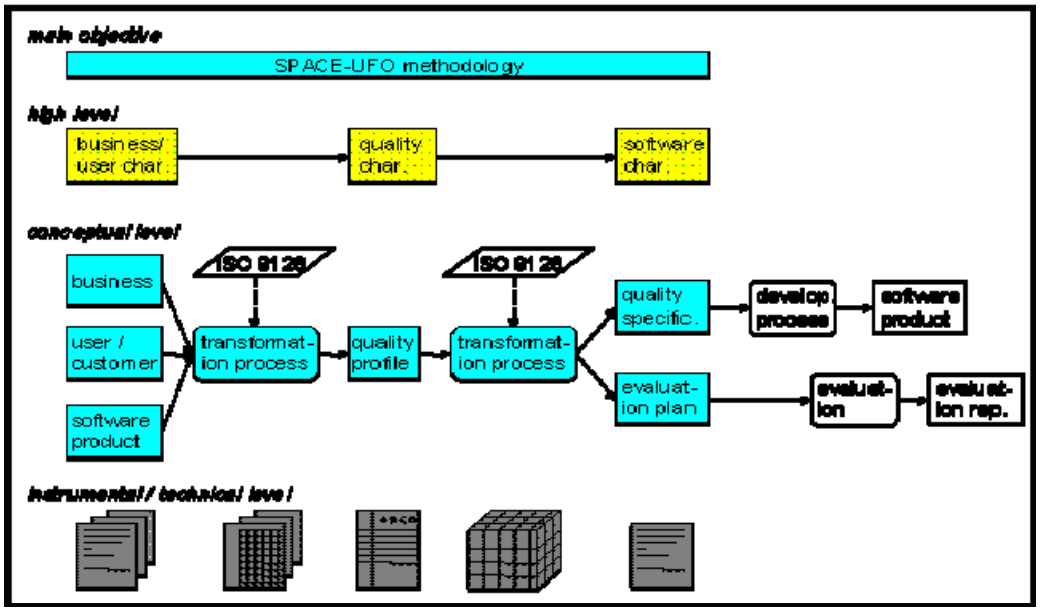


Figure 1.18 SPACE-UFO reference model
Extracted from Punter et al., (1997)

1.4.1.3 Analysis and discussion of the method

This methodology is based on building a quality profile which determines the quality level of the software product. The quality profile is then based on the user's quality needs related to business aspects and the quality characteristics of the software product. The methodology seems to be suitable to establish the important quality characteristics of the software product but some questions could be addressed:

- At which point of the process of building the quality profile are important quality requirements identified? Which techniques or tools have been used to identify quality requirements?
- Are business aspects well modeled to identify quality needs in a structured way?
- Are conflicts between quality characteristics resolved in a consistent and complete manner?
- Is there a way to retrace quality requirements or to manage their changes when they happen?

Table 1.3 summarizes the strengths and weaknesses of this method.

Table 1.3 Strengths and weaknesses of Space-UFO method
Extracted from Punter et al., (1997)

Space UFO METHOD	
Strengths	**Weaknesses**
1. Quality characteristics are extracted from the context in which the product is supposed to be used; 2. Quality needs of the stakeholders (users, customers and managers) are identified from the different business aspects; 3. Uses ISO/IEC 9126 to specify quality characteristics.	1. This method needs to structured & practical mechanisms to (Punter et al., 1997): a) Define the relationship between the quality characteristics of the product and the business characteristics and b) Specify and quantify quality characteristics: how to establish linkage between quality characteristics and their associated measures? 2. It is not mentioned anywhere in this method how to define "quality in use"; 3. It is not focused on the mapping activities of quality engineering instruments with the product definition phase at early requirements stage.7

1.4.2 MOQARE (Misuse-Oriented QuAlity Requirements Engineering) method

1.4.2.1 Description of the method

MOQARE (Hermann et al., 2007a.) is developed to explore quality requirements. The aim of MOQARE is to support intuitive and systematic identification of quality requirements. This method was developed by integrating and adapting concepts from other methods (like Misuse Cases) and provides a general conceptual model of quality requirements and a checklist-based process for deriving them in a top down fashion. This derivation starts from business goals and vague quality requirements and delivers detailed requirements. Relationships among these requirements are modeled in a Misuse Tree. The completeness criterion for the NFR is: each business goal must be linked to at least one business damage; each business damage must be linked to at least one quality deficiency.

Requirements addressed by this method are quality attributes (QAs), quality requirements (QRs). The tools/techniques supporting this method are the misuse case approach chosen as a basis for detailing QRs from business goals down to quality goals and further to detailed requirements (here called "countermeasures"). MOQARE identifies potential Misuse Cases with respect to all QAs and derives further requirements. The Misuse Cases method of exploring QRs is based on the general principle: an asset is to be protected from a threat, and to do so, countermeasures are defined. Figure 1.19 presents an overview of the MOQARE concepts and their relationships.

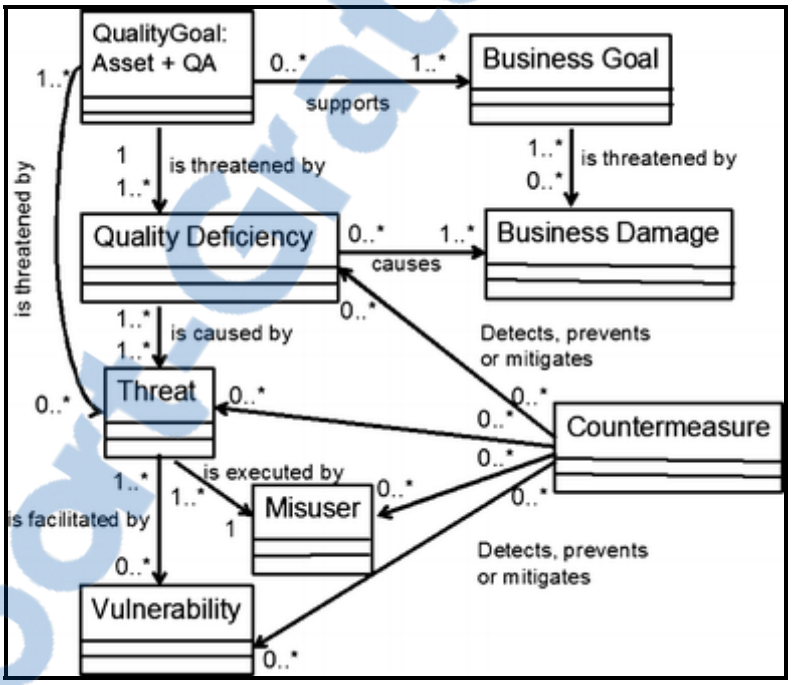


Figure 1.19 MOQARE concepts and their relationships
Extracted from Herrmann et al., (2007a)

Hermann (Hermann et al., 2007a) defined these concepts as follows:

- The *business goals* are supported by *quality goals* of the system. A quality goal is the combination of an asset plus a QA, and both are to be protected, like "integrity of the data". An *asset* can be any part of the system. The quality goals are high-level QRs.

- A *quality deficiency* means that the asset does not satisfy the QA. The quality deficiencies concretize how (when/where/how much) the system does not satisfy the QA. This non-compliance can be total or partial, permanent or temporary.
- A *threat* is an action (during system use, development, administration or maintenance) which causes a quality deficiency and consequently degrades the satisfaction of a quality goal. The threat is usually executed by a *misuser*, its driving force. Often, the threat is facilitated or even provoked by vulnerability.
- *Vulnerability* is a property of the system, either a flaw or a side-effect of an otherwise wanted property, if it is misused with respect to a quality goal.

1.4.2.2 Process of the method

The process model is presented in Figure 1.20 which describes MOQARE's general conceptual model of QRs and the checklist-based process for deriving them in a top-down fashion. The requirement elicitation is guided by a four-step process:

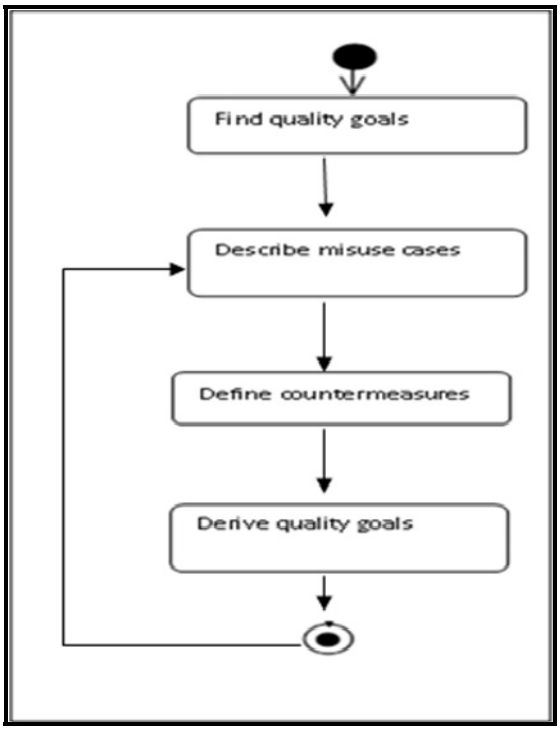


Figure 1.20 MOQARE process model

1. Find the quality goals (based on business goals, business damages, and quality deficiencies);
2. Describe Misuse Cases (including threat, misuser, vulnerabilities, and consequences);
3. Define countermeasures;
4. With countermeasures which are quality goals, re-start the cycle at step 2.

The MOQARE results can be presented in the form of a graph, a "Misuse Tree" (Figure 1.21). A Misuse Tree has the following levels, from top to bottom:

- Business goal: the cause of a system's development and use;
- Business damage: threat to business goals;
- Quality deficiency: cause damages;
- Quality goal: combination of an asset and a QA;
- Misuse Case : a whole misuse case scenario, including misuser, threat and consequences;
- Countermeasure, some of which are quality goals: prevents, mitigates or detects misuse.

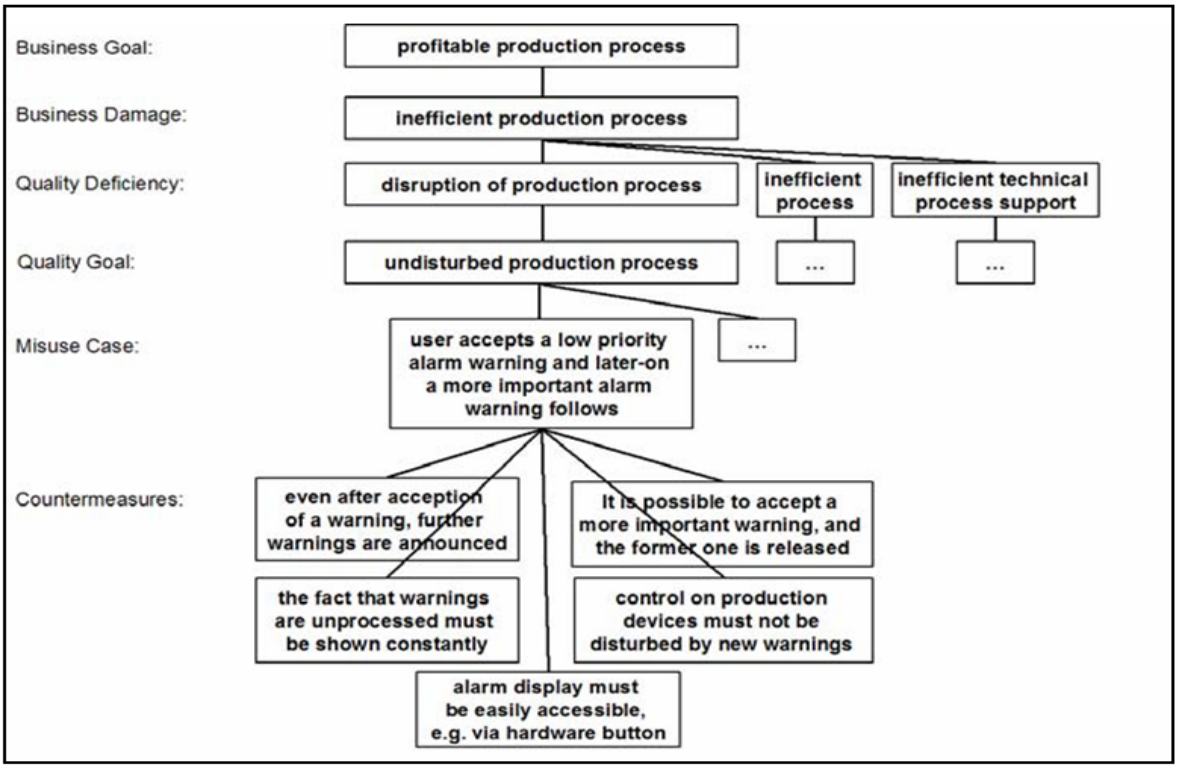


Figure 1.21 Misuse Tree for the wireless network system
Extracted from Herrmann et al., (2007b)

1.4.2.3 Analysis and discussion of the method

MOQARE supports intuitive and systematic identification of quality requirements. The input is a functional description or draft of a planned or existing system, business goals and quality goals. The output is a misuse tree. The method provides a systematic detailing of the NFR using defined concepts which are supported by a notation with a tree structure. MOQARE looks at quality deficiencies triggered by misuses in order to better understand what quality means to the stakeholders. The main contribution is support by the context-rich misuse case scenarios and the focus is on the business goals as main drivers of the system. The main quality issues captured by MOQARE need not to be measurable at an early stage. Metrics would only be emphasized as soon as they are needed to support quality assurance. However, there are some questions related to applicability of MOQARE method: are conflicts between QAs documented? How does one retrace QRs to their original requirements? And finally, for a complex system where the misuse tree gets big, is the MOQARE analysis time-consuming?

Table 1.4 summarizes the strengths and weaknesses of this method.

Table 1.4 Strengths and weaknesses of MOQARE
Extracted from Herrmann et al., (2007a and 2007b)

MOQARE method	
Strengths	**Weaknesses**
1. Based on business goals and focus on quality requirements which support business goals; 2. Support intuitive and systematic identification of QRs; 3. Provides a general conceptual model for QRs and a checklist-based for deriving them in a top down fashion; 4. Provides reuse of checklists; 5. Supported by the context-rich Misuse Case scenarios.	1. Vague NFRs are refined to FRs, or NFR but not measurable and quantifiable by metrics 2. Conflicts between quality concepts are not documented; 3. No direct integration of NFRs into the FRs documents and architectural options 4. Not yet applicable to all types of quality requirements; 5. Not yet proven its applicability in industry; 6. MOQARE analysis seems to be time consuming where the Misuse Tree gets too big and the system to analyze is too complex.

1.4.2.4 Suggestions

Suggestions have been made by authors (Hermann et al., 2007b.) to improve this method:

- The MOQARE process could include a final evaluating phase in which project specific knowledge is added to the checklists as additional items and also as a whole sub tree;
- Adopt an NFRs dependency graph analysis as an additional reusable artefact describing frequent QAs dependencies and their conflicts;
- Develop tools support to allow linking of NFRs to FRs and document their integration into the FRs documents.

1.4.3 ATAM (Architecture Tradeoff Analysis Method)

1.4.3.1 Description of the method

ATAM is an analysis method developed by the Software Engineering Institute at Carnegie Mellon University. The method is organized around business drivers and quality attributes goals and based on the extent of the architectural styles to determine quality attribute goals. Its purpose is to assess the consequences of architectural decisions in light of quality attributes requirements (Kazman et al., 2000). ATAM is most beneficial when done early in the software development life cycle when the cost of changing architectures is minimal.

ATAM is founded in three key concepts: quality attribute characterization, scenarios and the attribute-based architectural styles (Kazman et al., 2000). The scenario-based quality requirements elicitation is an important factor in applying this method.
Requirements addressed by this method are system quality attributes (Figure 1.22). Tools/techniques supporting this method are utility tree, scenarios and brainstorming.

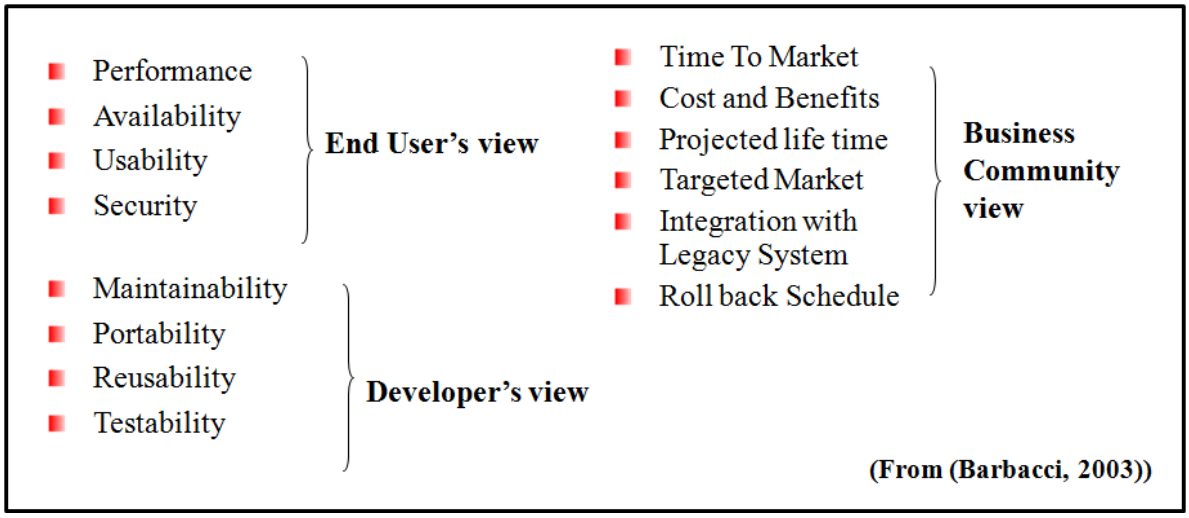


Figure 1.22 System quality attributes

1.4.3.2 Activities of the method

The ATAM process consists of gathering stakeholders together to identify the driving quality attributes from the business drivers and to create associated prioritized scenarios. These scenarios are then combined with architectural approaches and architectural decisions to identify trade-offs, sensitivity points, and risks (or non-risks).

1.4.3.3 Steps of the ATAM Process (Kazman et al., 2000)

The process model of ATAM is described in the following steps:

1. Present ATAM - Present the concept of ATAM to the stakeholders, and answer any questions about the process;
2. Present Business Drivers - Everyone in the process presents and evaluates the business drivers for the system in question;
3. Present the Architecture - The architect presents the high level architecture to the team with an 'appropriate level of detail';
4. Identify Architectural Approaches - Different architectural approaches to the system are presented and discussed by the team;
5. Generate a Quality Attribute Utility Tree - Define the core business and technical

requirements of the system, and map them to an appropriate architectural property and present a scenario for this given requirement;

6. Analyze architectural approaches - Analyze the scenarios, rating them by priority. The architecture is then evaluated against each scenario;
7. Brainstorm and prioritize scenarios - among the larger stakeholder group, present the current scenarios, and expand upon them;
8. Analyze architectural approaches - Perform step 6 again with the added knowledge of the larger stakeholder community;
9. Present results - provide all documentation to the stakeholders and write a report detailing this information along with any proposed mitigation strategies.

1.4.3.4 Analysis and discussion of the method

ATAM is a method for architecture evaluation which confirms that quality requirements were satisfied by the developed software architecture. ATAM evaluates architectures of multiple quality attributes, identifies critical architectural decisions that conflict among multiple quality attributes and resolves them. Quality requirements elicitation is the first step of ATAM where quality scenarios and requirements are gathered by interviewing the involved stakeholders of the software. However, the author mentioned (Lee et al., 2001) difficulty constructing any concrete quality scenarios. Reasons for the difficulties are: a) lack of consensus on the definition of quality attributes; b) biased viewpoints of some stakeholders; c) no systematic way to write scenarios and no metrics to evaluate architecture on multiple quality attributes scenarios. The proposed quality requirements elicitation strategy is represented by the following points:

1. Select the quality attributes;
2. Make a consensus on these quality attributes;
3. Develop scenario elicitation forms;
4. Select an appropriate measure for each quality attribute;
5. Decide priorities among the quality attributes.

Table 1.5 summarizes the strengths and weaknesses of this method.

Table 1.5 Strengths and weaknesses of ATAM method
Extracted from Kazman et al. (2000) and Lee et al., (2001)

ATAM method	
Strengths	**Weaknesses**
1. The leading method in the area of software architecture evaluation; 2. An interesting analysis method based on business drivers; 3. Focused on the stakeholder's scenarios, quality attribute characterization and quality attribute architectural styles.	1. Used at the architectural level, not requirement one; 2. Time consuming in writing scenarios and interviewing stakeholders; 3. Difficulty to understand terminologies related to quality attributes definitions 4. Unavailability of various stakeholders and no personal profiles of stakeholders; 5. No systematic way to write scenarios; 6. Scenarios have not metrics leading to difficulty in analyzing tradeoffs and evaluating architectures.

1.4.4 FDAF (Formal Design and Analysis Framework) method

1.4.4.1 Description of the method

FDAF is an aspect-oriented architectural approach proposed to solve the problem of systematically modeling and analyzing NFRs for software architecture (Dai et al., 2005). This approach allows design and analysis of NFRs for distributed real systems and helps to build NFRs aspects into software architecture involved in enterprise level goals. This approach is supported by a process providing a systematic modeling of NFRs properties (by extending UML with aspects) and their automated analysis (by using formal methods and their supporting tools). This process verifies that these NFRs have been met and allows one to decide how to reorganize architecture components affected by these NFRs. In FDAF, NFRs' properties are represented as aspects at the architectural level. An aspect repository is

provided to reuse predefined aspects. These aspects are integrated into the UML based architecture design model and analyzed automatically (by using translation algorithms) to formalize part of UML into formal languages.

Requirements addressed by this method are quality aspects. Tools/techniques supporting this method are: UML model, Aspect Oriented Paradigm, Formal methods and their supporting tools.

1.4.4.2 Activities of this method

The process model is presented in Fig 1.23 where a UML aspect-oriented design model is created, formalized, analyzed and iteratively refined according to analysis results provided for particular aspects. Activities of FDAF are presented below (Dai et al., 2005).

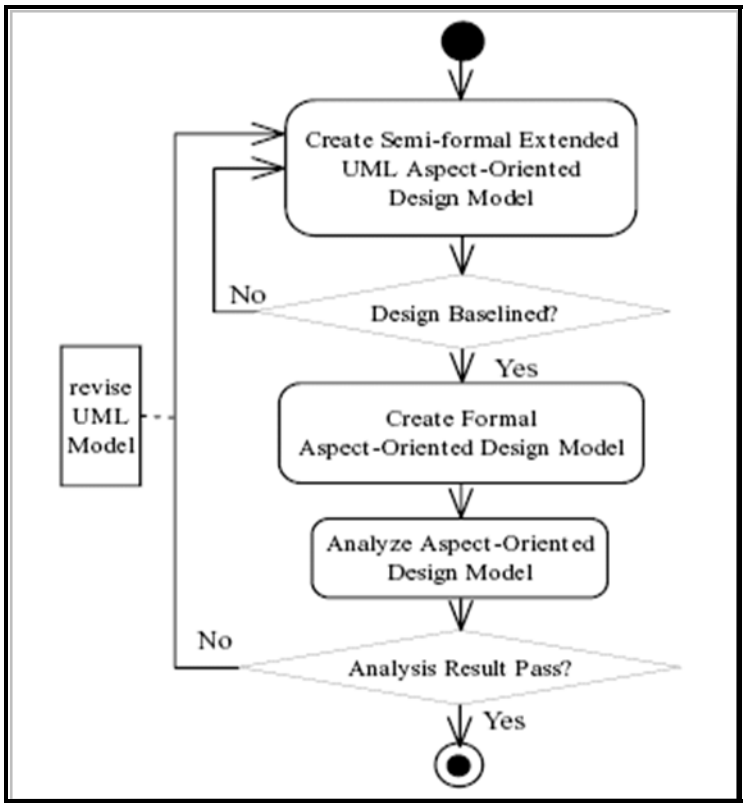


Figure 1.23 FDAF process model
Extracted from Dai et al., (2005)

- **Create a semi formal Extended UML Aspect Oriented Design Model:** NFRs of the system are represented as aspects. A parallelogram notation is used to capture aspects in the UML design model where UML can be extended by stereotypes, tagged values and constraints.
- **Create a formal Aspect Oriented Model:** A suitable formal language associated with the aspect oriented UML design model is selected and translated into a set of formal models.
- **Analyze the formal Aspect Oriented Model:** The set of formal models is analyzed using existing tool in support (as Promella and Rapide) of the formal languages.

1.4.4.3 Analysis and discussion of the method

FDAF is an interesting method used to create architecture designs with NFRs aspects that cannot be described in the real time version of UML. The major contribution of FDAF is that it integrates the semi-formal UML with formal methods into an aspect oriented framework. In fact, the parallelogram notation is used to present aspect information. The aspect model is based on one specific aspect which makes it simpler than a traditional mixed model. However, formal methods are limited by their analysis tools in different areas. For instance: a lack of modeling constructs to support the description of a component's behavior and connections; difficulty to obtain useful information from the raw data as the number of simulated events increases; restriction of modeled systems by mathematical assumptions and time consumption related to analysis of NFRs aspects.

Table 1.6 summarizes the strengths and weaknesses of FDAF method when applied to define and analyze the three quality aspects (Dai, 2005) and (Dai et al., 2003, 2005 and 2006): performance response time aspect analyzed with Rapide tool, performance resource utilization aspect analyzed with Armani tool and the RBAC (Role Based Access Control) security aspect analyzed with Alloy tool.

Table 1.6 Strengths and weaknesses of FDAF method
Extracted from Dai (2005) and Dai et al., (2003, 2005 and 2006)

FDAF METHOD	
Strengths	**Weaknesses**
1. The FDAF performance aspect analysis helps to create architecture design that cannot be described in the real time version of UML; 2. Rapide's analysis tool supports architects with detailed analysis of the system's behaviour simulation at the architectural level and detects uninspected activities; 3. Rapide is of great help for architects and designers to identify early problems and to refine the architecture design iteratively; 4. The FDAF resource utilization aspect analysis provides architects with detailed analysis information about which component is the bottleneck (overloaded and busy all the time) and refine the UML architecture to meet the NFRs; 5. The FDAF security aspect analysis allows detecting inconsistency of the multiple system security policies early in the design.	1. There is no identification step of the NFRs aspects by the framework; 2. Limitations of the Alloy's analysis tool in this area: it doesn't provide modeling constructs to support the description of component's behaviour and connections; 3. Analysis is time consuming; 4. There is no mention where the quality standard ISO/IEC 9126 has been used; 5. Limitations of the Rapide's analysis tool and difficulty to obtain useful information from the raw data (response time analysis results presented in the graphical browser) as the number of simulated events increases; 6. Limitations of the Armani's analysis tool in this area: the mathematical assumptions restrict the systems they are modeled. For example assumptions that all components are executing sequentially are not applicable to systems where components are executing in a parallel way; 7. The queuing network analysis is not applicable to other architectural styles (pipe and filter and layer architecture); 8. The Armani tool does not calculate automatically the property "sOverloaded" instead it allows changes to it.

1.4.5 Method "Requirement model for quality attributes"

1.4.5.1 Description of the method

This method defines a process to identify and specify quality attributes that crosscut requirements and to integrate them into the functional requirements at an early stage of the software development process (Brito et al., 2002):

1. Proposes a template to specify quality attributes at the requirement stage;
2. Extends "Use Cases" and sequences diagrams (Jacobson et al., 1992) to specify integration of quality attributes with functional requirements.

1.4.5.2 Activities of the method

The process model is compatible with UML formalism (Jacobson et al., 1998) and is composed of three important activities (Figure 1.24):

1. Identification of system requirements and selection of quality attributes relevant to the stakeholder's requirements and application domain from those requirements;
2. Specification of requirements:
   - Specify functional requirements by using "Use Case" based approach;
   - Describe quality attributes by using templates and specify quality attributes crosscutting functional requirements;
3. Integration of crosscutting quality attributes with functional requirements.

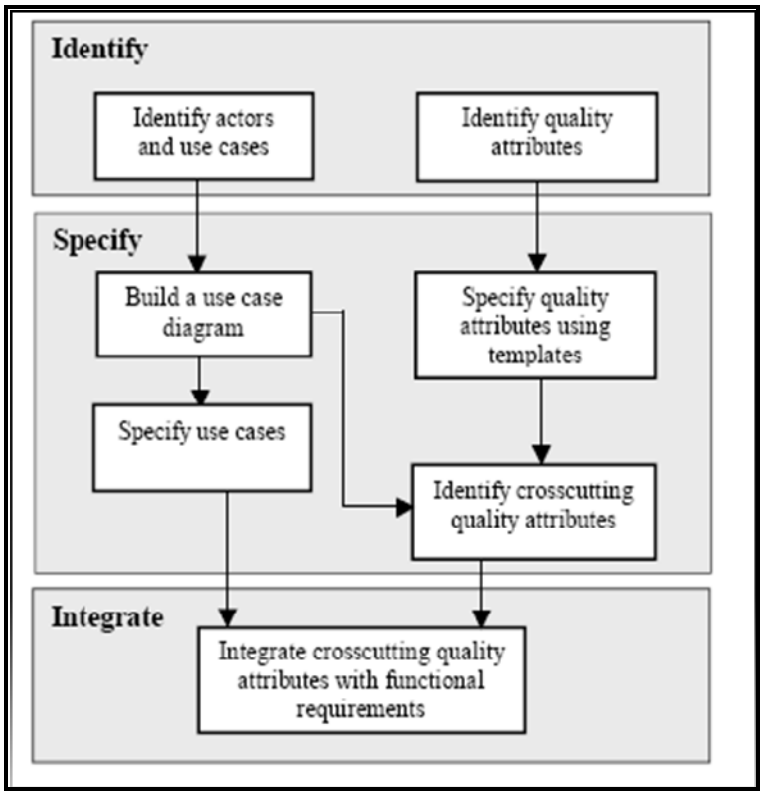


Figure 1.24 Requirements model for quality attributes
Extracted from Brito (2002)

Requirements addressed by this method are: functional and quality. Quality requirements are specified as "quality attributes" and are defined as "global properties of a system, assumptions, constraints or goals of stakeholders". The quality model used by this method is a template for describing quality attributes. Tools/techniques supporting the method are Use Case approaches (UML model, sequence & class diagrams) for specifying functional requirements and templates for describing quality attributes.

1.4.5.3 Analysis and discussion of the method

The method "Requirement model for quality attributes" defines a process to identify and specify quality attributes that crosscut requirements including their integration with functional requirements.

The strengths of the method (Araujo et al., 2002 and 2003) and (Brito, 2002) are:

1. It proposes a new concept: “aspect-oriented paradigm", to integrate quality requirements (non functional requirements) with the functional requirements (Araujo et al., 2002 and 2003).
2. This method investigates other approaches such as ATAM (Architecture Tradeoff Analysis Method), composition patterns and goal oriented requirements engineering related to quality attributes and crosscutting concerns.
3. Using a template for describing quality attributes is interesting in the sense that knowledge about these attributes is collected (source, focus, decomposition, influence, requirements describing them, and their contribution to other attributes).

However somr drawbacks are identified:

1. it is not specified anywhere how to identify these quality attributes from system and user requirements and how to select them according to the application domain and stakeholders (Djouab and Suryn, 2006).
2. In addition, it is not indicated in the template how quality attributes are derived from quality requirements and how they are retraced to these quality requirements.
3. Finally, It is not specified how ISO/IEC 9126 is used to specify quality characteristics and sub-characteristics.

1.4.6 IESE NFR method

1.4.6.1 Description of the method

IESE NFR is a systematic experience-based approach which elicits documents and analyses Non-functional Requirements (NFRs) of embedded systems. Its objective is to achieve a minimal and sufficient set of measurable and traceable NFRs (Doerr et al., 2005). IESE NFR has been introduced to palliate the drawbacks of other approaches which lack systematic guidance on how to use them and to end up with measurable NFRs. IESE NFR distinguishes

between quality attributes (QAs) and NFRs where QAs are captured in quality models and NFRs are captured in templates. IESE NFR defines QAs as “QA is a non-functional characteristic of a system, user task, system task, or organization. An NFR describes a certain value of a QA that should be achieved in a specific project” (Doerr et al., 2005).

The IESE NFR methodology has been used to elicit usability requirements in concert with supplementary requirements related to “Use Case” approach and high level architecture (Kerkow et al., 2003). Kerkow shows how quality aspects contribute to architectural design. The methodology uses a quality model (QM) (Figure 1.27) and quality attribute (QA) types to capture knowledge on NFRs and a template for capturing specific NFRs. In addition, checklists are used to elicit NFRs in concert with user models, Use Cases and architecture.

1.4.6.2 Activities of this method

IESE NFR method is organized around stakeholder workshops to select and tailor quality models and to use these models to elicit and document the NFRs. In fact, the method starts by prioritizing the high level QAs most important to the project and by selecting the quality models associated to these QAs. These selected quality models are tailored in workshops to the needs of the project. Checklists and templates are derived from the quality model to be used (in workshops) for the elicitation process. Dependencies between QAs (general and the lowest level) in the quality models are included in the checklists and used to identify NFRs and conflicts among them. The process of IESE NFR is organized around 2 basic steps (Figure 1.25): tailoring the quality model and elicitation process.

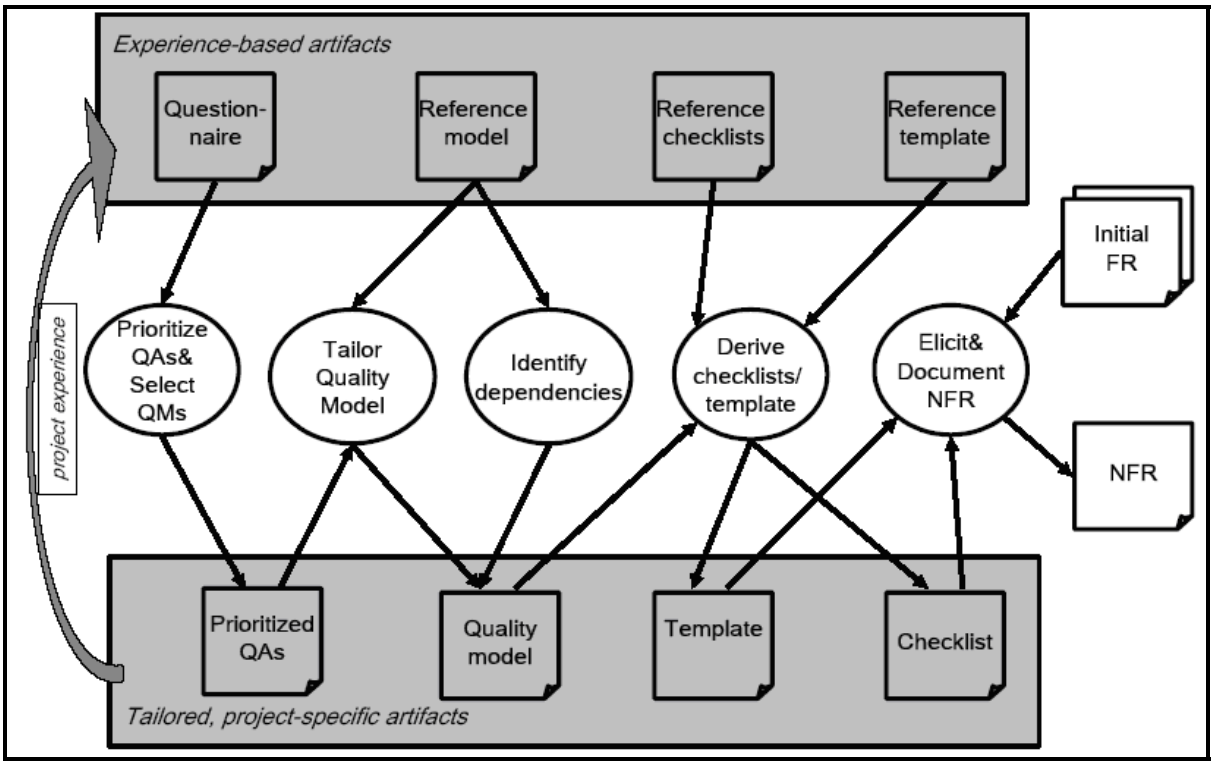


Figure 1.25 IESE NFR process
Extracted from Doerr et al., (2005)

Tailoring the quality model: where the experience based reference model is tailored to the need of the client's project (Figures 1.26 and 1.27). This process produces checklists and templates for use in the next process. The figures 1.28, 1.39 and 1.30 show examples of the tailoring process for the Tetris game.

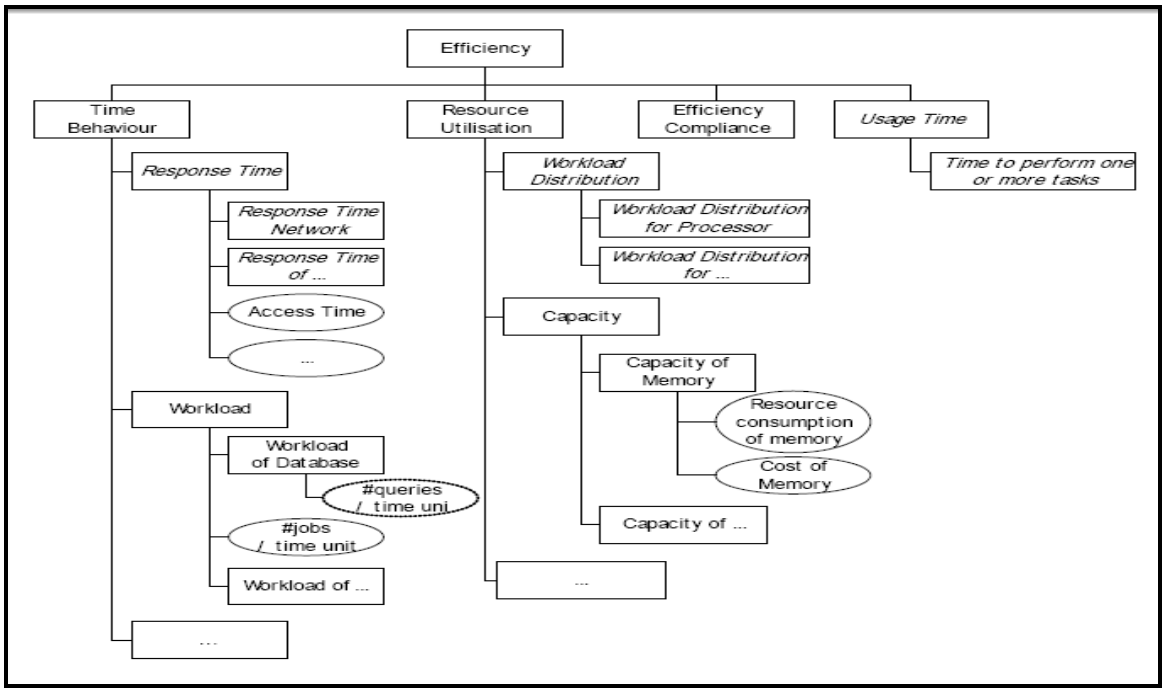


Figure 1.26 Quality reference model for Efficiency
Extracted from Doerr et al., (2005) and Kerkow et al., (2003)

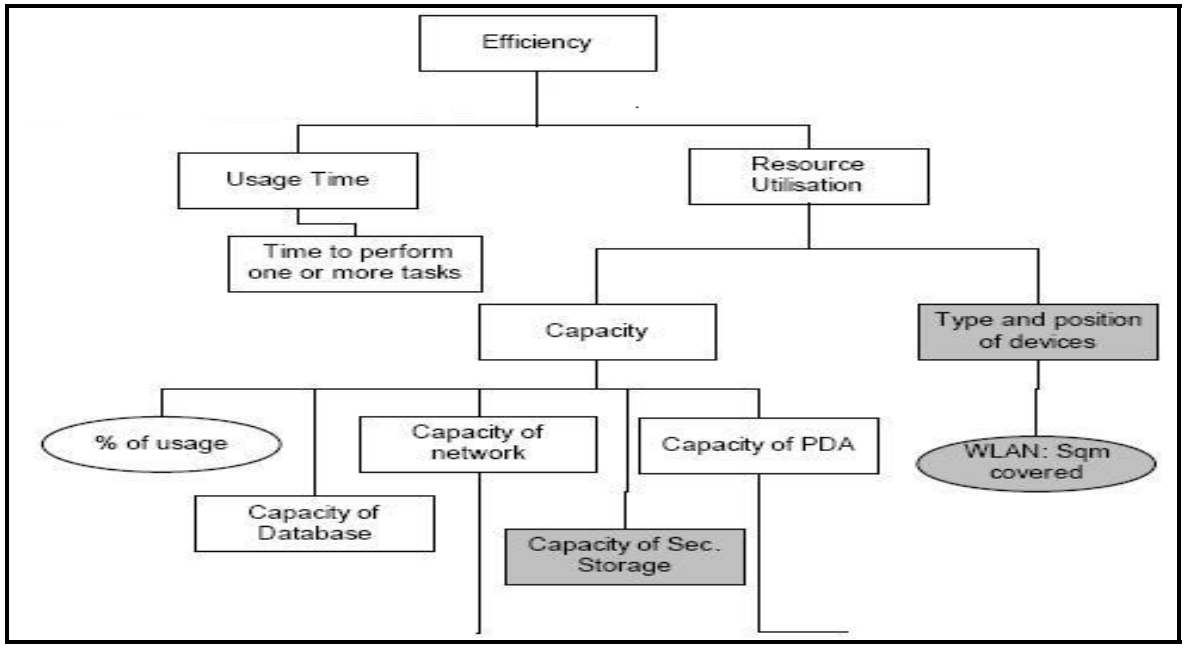


Figure 1.27 Tailored QM for Efficiency
Extracted from Doerr et al., (2005) and Kerkow et al., (2003)

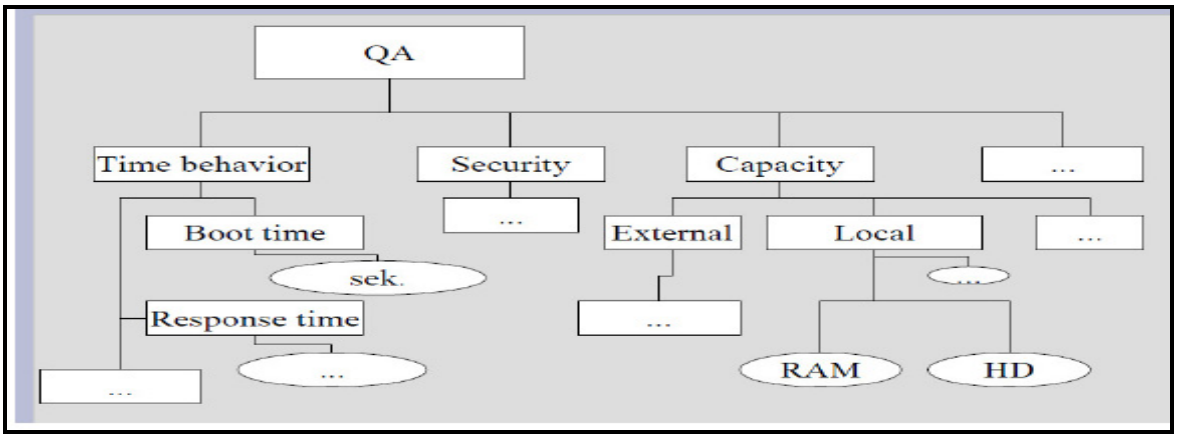


Figure 1.28 Tailoring process example of game Tetris
Extracted from Herrmann et al., (2007b)

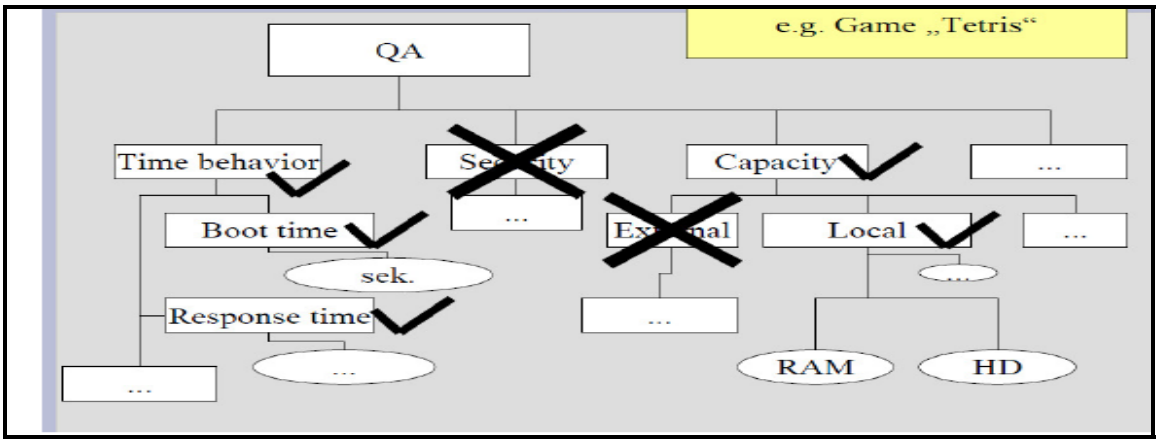


Figure 1.29 Tailoring process example of game Tetris
Extracted from Herrmann et al., (2007b)

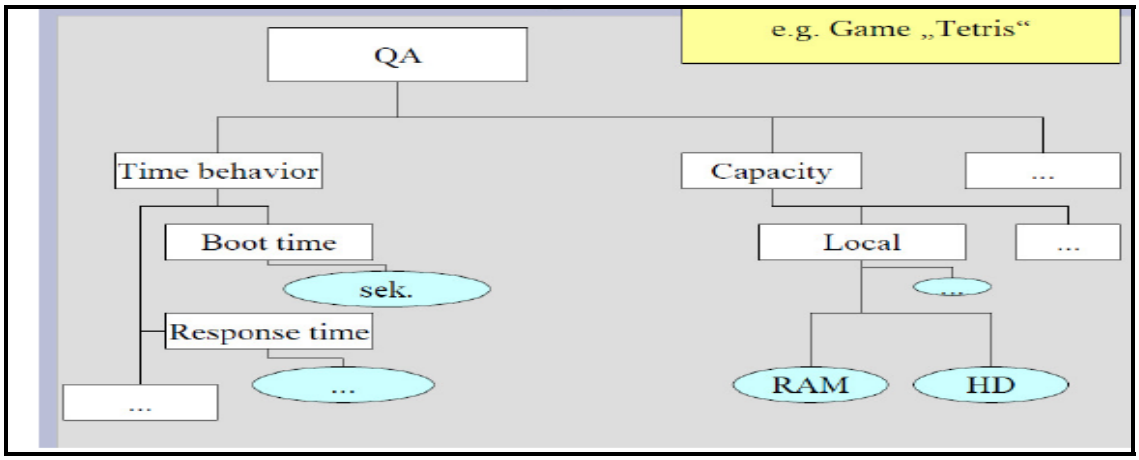


Figure 1.30 Tailoring process example of game Tetris
Extracted from Herrmann et al., (2007b)

Elicitation process: Based upon the previous created artifacts, the different types of activities that formulate the NFRs are defined (organizational, user task, system task and system). These NFRs are consolidated to be analyzed for possible conflicts.
Activities of the elicitation process are:

- Elicit organizational NFRs; elicit NFRs that constrain QAs of the organization;
- Elicit user task NFRs; elicit NFRs that constrain QAs of user tasks;
- Elicit system task NFRs; NFRs that constrain QAs of system tasks;
- Elicit system NFRs; elicit NFRs that constrain QAs of the system and subsystems;
- Consolidate; QAs are analyzed for conflicts and NFRs that constrain different QAs are validated according to dependencies documented within the quality model.

The checklist gives a means to identify these conflicts and a means to solve them. The process is based on the following artifacts: Prioritized questionnaire; user model; system functionality and physical architecture. The figures 1.32, 1.33 and 1.34 show examples of the elicitation process for the Tetris game.

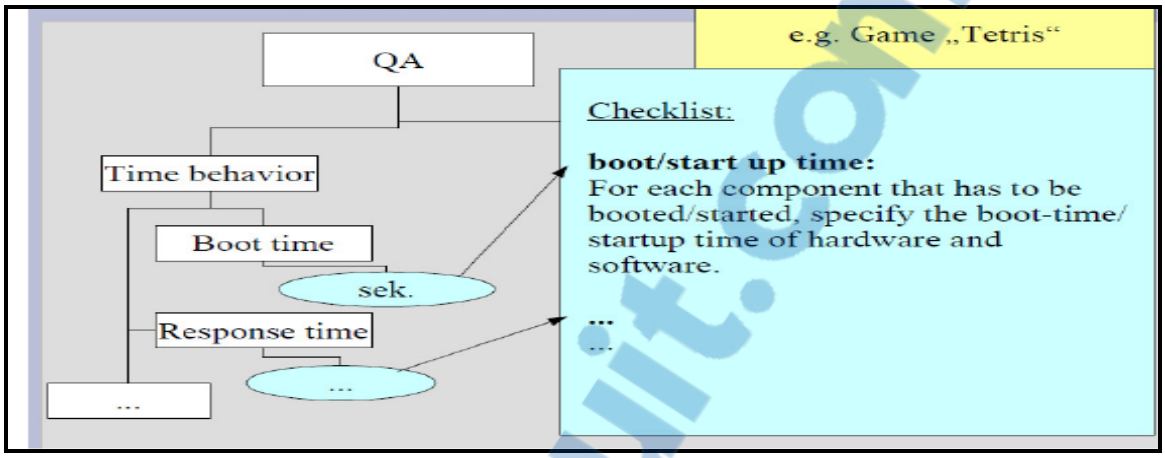


Figure 1.31 Elicitation process example of game Tetris
Extracted from Herrmann et al., (2007b)

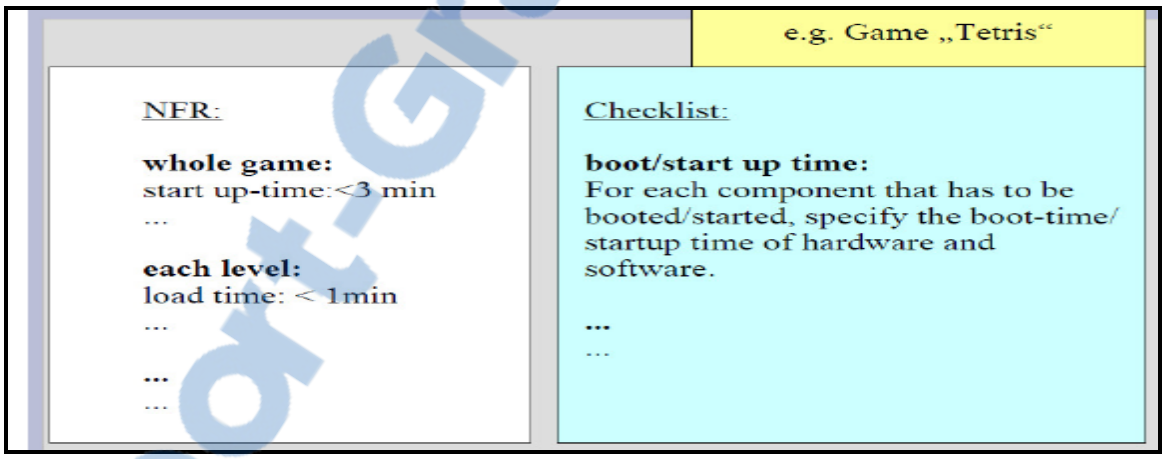


Figure 1.32 Elicitation process example of game Tetris
Extracted from Herrmann et al., (2007b)

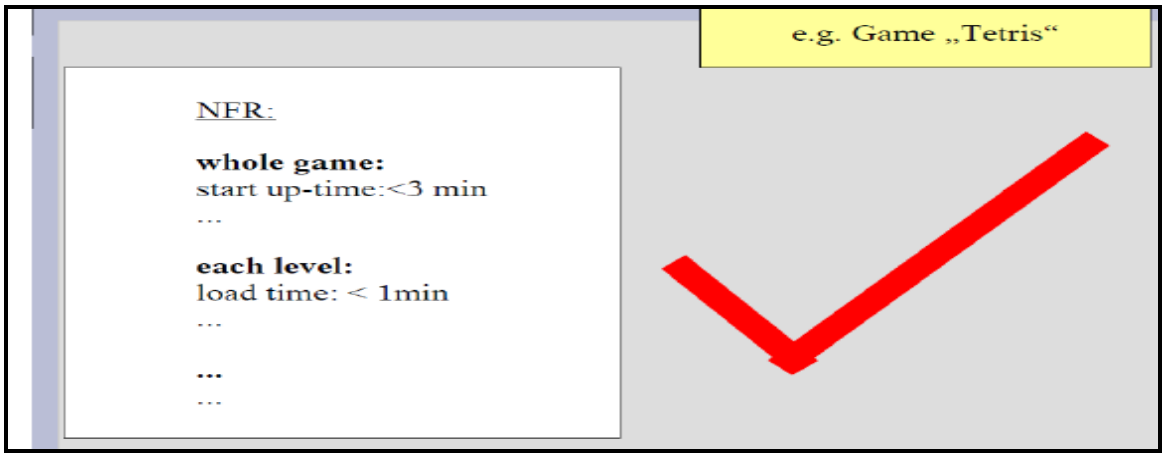


Figure 1.33 Elicitation process example of game Tetris
Extracted from Herrmann et al., (2007b)

1.4.6.2 Analysis and discussion of the method

IESE NFR method tried to achieve a complete and focused set of measurable and traceable quality aspects at an early stage. It provides structured guidance to elicit and document NFRs supported by the prioritized questionnaire of QAs, the tailoring process of the QM and the derived checklists and templates. The main contribution of this method is to provide guidance during the elicitation process. Hence, IESE NFR is suitable to deal with quality requirements at early stages because it deals with multiple NFRs (high and lowest levels) and is based on ISO/IEC 9126. However, some limitations remain:

- The method is restricted to embedded systems where NFRs are dependent on functional requirements and architectural options and where NFRs should be clarified in the subsequent phases of the development process;
- How can one retrace quality attributes to their original quality requirements?
- What should be done with NFRs not satisfied?
- The method will become difficult to use with the complexity of the quality model.

Table 1.7 summarizes the strengths and weaknesses of the method when applied in industry (three case studies) (Doerr et al., 2005) and dealing with security, efficiency, reliability and maintainability attributes.

Table 1.7 Strengths and weaknesses of IESE NFR method
Extracted from Doerr et al., (2003 and 2005) and Kerkow et al., (2003)

IESE NFR METHOD	
Strengths	**Weaknesses**
1. A systematic approach which led to structured, correct, complete and measurable NFRs; 2. Identifies early conflicting requirements with the use of the analysis dependency; 3. Enhances communication between stakeholders (requirements engineer, developer and customer); 4. Ability to elicit several quality attributes; 5. NFRs are in almost cases measurable.	1. How are conflicts among quality attributes resolved? 2. How to maintain the quality model with the growth of the dependency graph? 3. The dependency graph is used to represent dependencies between quality attributes. The graph is not used to capture NFRs (they are placed in the requirements documents template); 4. Size of checklist will be large with the growth of the conditions and alternatives sections; 5. The experience based artifacts (models, checklists and templates) have to be maintained to be used efficiently. 6. Much time is spent to resolve terminologies problems during workshop sessions;

1.4.7 Soft goal notation of the Chung NFR Framework

1.4.7.1 Description of the framework

The framework is a process-oriented approach addressing non functional requirements (NFRs) (Chung et al., 1994, 1995 and 2000). It uses a goal graph structure to record and structure NFRs, design alternatives, decisions and rationale. All of these concepts are treated uniformly as goals (denoted by nodes) and related to one another via links. The framework documents NFRs with soft-goal notation (Figure 1.35) where elements of the goal graph are: a) quality attributes/NFRs (represented as clouds), b) operationalizations (represented in bold clouds) which indicate how the NFR is achieved and c) relationships divided into refinement (represented by "And" and "Or") and contribution (represented by positive contribution "Make" and negative contribution "Break"). To satisfy NFRs goals, the developer considers

design alternatives called “satisfying goals” along with their tradeoffs, refines them, makes selections and justifies them by recording design rationale called “argumentation goals”.

Development knowledge about specific NFRs is to be taken from the literature and industrial experience and captured as *methods,* which are then presented for reuse to help the developer generate new goals and links. For example, techniques can be incorporated from security evaluation criteria, performance responsiveness principles, and accuracy concepts (Chung et al., 1995).

Non-functional requirements are systematically integrated into the development. They are represented as potentially conflicting or synergistic *goals.*
To deal with NFRs, there is a need to:

- Consider key domain characteristics;
- Capture NFR-specific concepts;
- Detect defects.

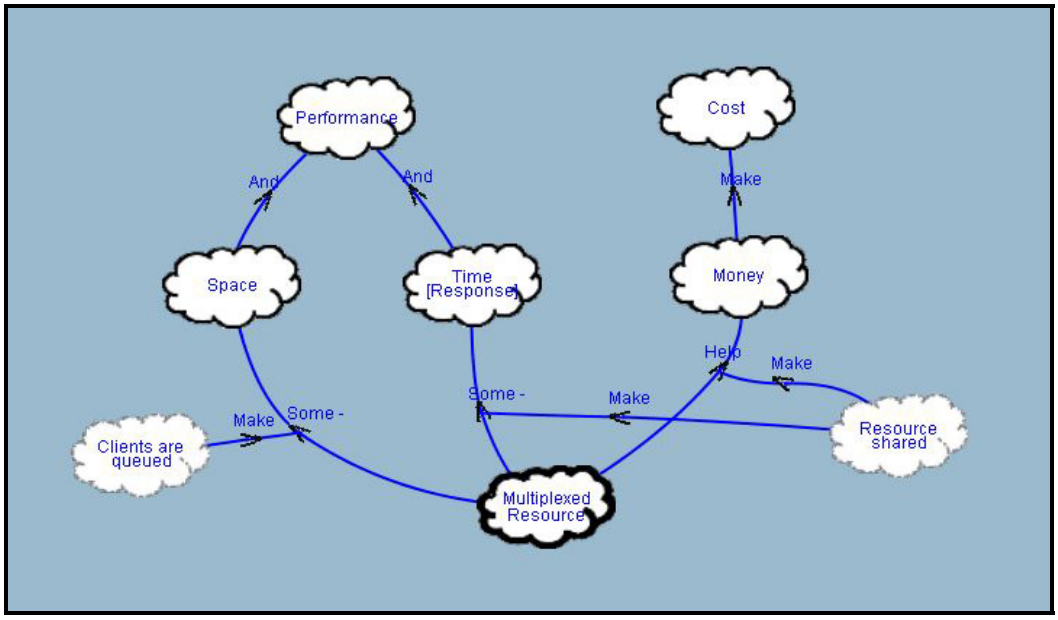


Figure 1.34 Soft-goal notation example
Extracted from Chung et al., (1995)

Requirements addressed by the framework are non functional requirements (NFRs) or soft goals. Tools/techniques supporting this method are goal graph structures, catalogues of refinement methods and interdependencies analysis among NFRs.

1.4.7.2 Activities of the framework

The process model is presented in Figure 1.36

Step1: knowledge acquisition

1. **knowledge acquisition specific to NFRs**
   - ❖Obtain knowledge about the particular type of requirement from industrial experience and academia literature
   - ❖Encode this knowledge and catalogue it to produce a terminology for the quality requirements, a list of generic techniques and their tradeoffs and interactions.

2. **Acquisition of domain knowledge**
   - ❖Obtain documents from the organizations whose systems to be studied. They can be: system development documents; partial schema information; workload information and information on the organization(its operations and its system)
   - ❖Conduct an initial review of the documents to understand the domain characteristics, obtain the NFR related information and get idea of some of the main requirements.
   - ❖Argument information on schemas and requirements

Step2: Application of the NFR framework

1. **Identification of NFRs related concepts**
   - ❖Identify important NFRs goals
   - ❖Identify development techniques
   - ❖Identification of design rationale

2. **Linking NFRs related concepts**
   - ❖Refine, clarify and relate quality goals
   - ❖Identify critical NFRs goals
   - ❖Provide design rationale
   - ❖Assess goal achievement

(from (Chung et al., 1995))

Figure 1.35 Framework model

1.4.7.3 Analysis and discussion of the method

The *NFR-Framework* is based on a *process-oriented approach* to deal with NFRs. During the software development process, this framework allows treating NFRs as potentially conflicting goals to achieve. Development alternatives which could meet the stated NFRs are considered and design tradeoffs are examined. The design decisions related to NFRs are

justified according to the needs of the intended application domain. The framework has been evaluated from three viewpoint perspectives: developers, experts and application domain (Chung et al., 1995).

1.4.7.4 Perspective of developers

- Framework observations: explicit expression of an initial set of NFRs as goals improved awareness and led to systematic development. When conflicts and synergy among the NFRs goals are explicitly described, allowing consideration of design tradeoffs to satisfy NFRs goals;
- Catalogues of methods enable one to capture the large number of NFRs-specific concepts and their associated techniques;
- Goal graph structures are important as a record of initial development and for long term review and maintenance of systems;
- Defect detection observations dealing with NFRs involve repeated clarification of goals, addition of missing details, detection of goal graph synergy and conflict.

1.4.7.5 Perspective of domain experts

- Framework is helpful in the broad domain studied;
- The cataloguing of development techniques and NFRs-specific knowledge would be helpful;
- The goal graph structure and their components were helpful.

1.4.7.6 Application domain perspective

Application of the framework did not correspond to the domain because of the lack of knowledge about the domain, its priorities and terminology (lack of contact with domain people during the study).

In summary, important framework findings are a process oriented approach, goal graph structures, formality, tradeoffs and delivery of the main requirements. Table 1.8 summarizes the strengths and weaknesses of the framework.

Table 1.8 Strengths and weaknesses of Chung framework
Extracted from Chung et al., (1994 and 1995)

Chung framework	
Strengths	**Weaknesses**
1. Applicable to all types of quality requirements; 2. Structure and record NFRs, design alternatives, decisions and rationale in a goal graph structure; 3. Provides catalogues of refinement methods.	1. Knowledge on conflicts detection with functional requirements is not collected; 2. Elicited NFRs are not integrated in the requirements specification document; 3. Focused on documentation and negotiation of QRs and not their elicitation from business goals; 4. Some NFRs stated in quantitative terms are not supported by the taxonomy of the NFR framework; 5. Goal graph structures would be larger for complex systems; 7. New decomposition methods would be provided to bridge automatically the gap between the new NFRs and the given satisfying goals; 8. There is a comprehensibility limit in understanding the meaning of arguments; 9. Improvements in naming and presentation are needed to increase understandability; 10. Lack of consultation with domain people during the study left gaps in the domain knowledge.

1.4.8 Prometheus Method to model quality in SPL (Software Product Lines)

1.4.8.1 Description of the method

Authors Punter (Punter et al., 2002 and Trendowicz et al., 2003) try to combine several methodologies to model and evaluate the quality of software products early in the

development process. Prometheus is an example of such research. This approach describes a method to modeling NFRs (Non-functional requirements) by using flexible, reusable and transparent quality models. Prometheus combines 3 methodologies to model and evaluate the quality of a software product:

1. The SQUID approach (Kitchenham et al., 1997);
2. The BBNs (Bayesian Belief Networks) probabilistic concept (Fenton et al., 2002);
3. The GQM (Goal Question Metric) concepts (Gray et al., 1997, Birk et al., 1998, Fuggetta et al., 1998 and Solingen et al, 1999a).

1.4.8.2 Activities of the method

The definition process for these quality requirements is composed of three phases:

1. **Requirements specification phase:** during this phase a quality model is developed. Activities to define quality requirements are listed below (Figure 1.37):
   a. Define quality goals**:** quality goals are defined by the system users and other stakeholders by applying the MGT (Measurement Goal Template);
   b. Specify quality characteristics (content of model): describes the refinement of quality goals into quality characteristics and sub characteristics;
   c. Specify relationships (structure of model): here, two types of relationships are defined: **decomposition,** which specifies decomposition of high quality characteristics into detailed sub characteristics, and **influence**, that defines which sub characteristic influences the value of other characteristics;
   d. Review the model: the model is reviewed according to the implementation feasibility;
   e. Operationalize the model: the model is quantified (characteristics and relationships) and qualified by applying the BBN technique (Bayesian Belief Network).

2. **Application phase:** During this phase, the model is used to evaluate the requirements;

3. **Packaging phase:** Information on acquired experience during application of the model is collected in order to improve that information and to reuse it in other projects.

Requirements addressed by this method are quality goals and characteristics of the software product. Various quality models are combined and used by this method. Tools/techniques supporting this method are: GQM (Goal Question Metric) to define quality goals, interviews and questionnaires with domain experts to refine quality goals into quality characteristics and sub characteristics, SQUID tool for modeling and evaluating software quality and BBN technique for quantification of relationships as well as for the integration of quantitative and qualitative data within the quality model.

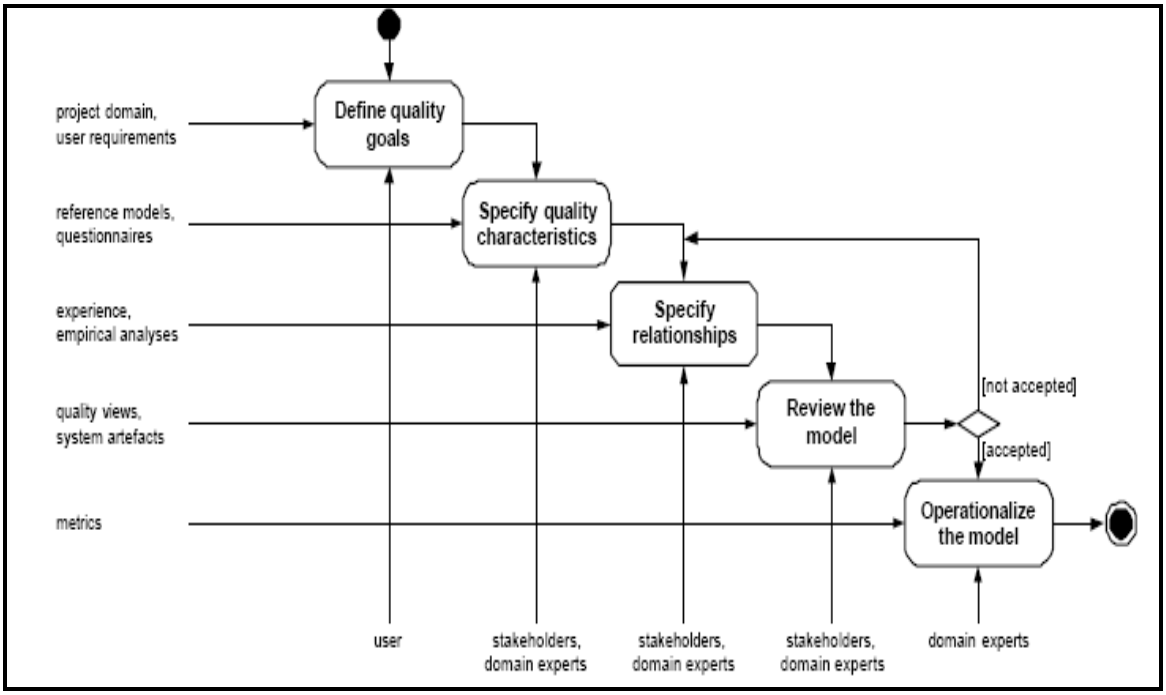


Figure 1.36 Activities during the specification phase of the Prometheus method
Extracted from Trendowicz et al., (2003)

1.4.8.3 Analysis and discussion of the method

As mentioned before, Prometheus method enables one to start quality evaluation early in the development process. It uses GQM method to define quality goals. The goal formulation is conducted iteratively and serves as a baseline for the evaluation step. Goals are defined by system users and other stakeholders related to the project who are involved in acceptation of the evaluation. This method defines its own quality model by organizing interview sessions

with domain experts who contribute in defining quality goals and quality characteristics. Sessions are supported by techniques such as questionnaires, case studies and existing quality models.

Prometheus is an interesting method since it gives the main activities for building the quality model for the project domain and has the following advantages:

1. Starting quality evaluation early in the development process;
2. Learning effectively across several product variances/releases;
3. Integrating quantitative (measured based) and qualitative approaches;
4. Combining different contexts of software quality individual views (as developers, users) and evaluation objects (processes, products, resources);
5. Applicable across different companies, to any project and incorporates views of all relevant project stakeholders;
6. Reuse of quality experience packaged in existing quality models across other projects. It also supports the reuse of measurement data as well as quality characteristics and their relationships;
7. Refining the quality model through subsequent projects.

However, this method does not use the ISO/IEC 9126 as a quality model and it does not indicate how quality requirements are extracted from quality goals and how they are specified (Djouab and Suryn, 2006). Further, application of Prometheus for quality modeling also faces problems like the huge effort needed to initialize the BBN quality model with expert's knowledge, serious limitations of the BBN network in size and structure (decomposition level equal to 2), computation complexity of the BBN network linked with the exponential growth of the number of probabilities and the size of tables, and as reported by Fenton, impossibility to assess accuracy of the quality model due to the great amount of data required to make precise predictions of the quality characteristics values (Djouab and Suryn, 2007a). Table 1.9 summarizes the strengths and weaknesses of this method.

Table 1.9 Strengths and weaknesses of Prometheus method
Extracted Trendowicz et al., (2003); Empress, (2004); (Gray et al., (1997); Birk et al., (1998); Fuggetta et al., (1998) and Solingen et al, (1999a)

Prometheus METHOD	
Strengths	**Weaknesses**
1. It gave a detailed description of software product quality requirements definition activities; 2. Combines both subjective probabilities (from domain experts) with probabilities based on objective measured data; 3. BBN provides a transparent quality model (in structure and content); 4. Provides easy learning of complex quality dependencies (conflicts ans redundancies); 5. Easy to be modified and to be applied in similar software projects; 6. Combines different kinds of data and facilitates merging more than one quality view in one model; 7. Specifies quality attributes and helps understand the relationship types among them (redundancies, contradictions); 8. Refines probabilities during the development process 9. Predicts missing data; 10. Supported by automatic tools (Analytica, Hugin, Netica, MSBNx);	1. There is no mention how to identify QRs from quality goals (defined by GQM method); 2. It did not use ISO/IEC9126 as quality standard; 3. There is no mention how to specify QRs; 4. The quality model is build for a specified software (embedded) and particular application domain; 5. Output of the Bayesian quality model is a probability of a value of the quality characteristic instead of the value itself. 6. Initial BBN quality model requires much effort from the experts to set up the node probability tables; 7. Limitations on the size of the BBN: the number of cells of a given BBN network augments exponentially with the growth of the number of variables and relationships; 8. Limitations on the structure of the BBN network : maximal number of parents limited to 2; 9. Exponential growth of probabilities requires more computational power to re-calculate the network; 10. The cost of the relationships quantification (in the decomposition tree combined with the quality model) may increase if each characteristic will be influenced by (or decomposed to) more than 3 sub characteristics; 11. Problem of definition of the BBN parameters (conditional probabilities); 12. Quality model developed with GQM is specific to the project domain & the characteristics/sub characteristics obtained during the refinement process are not in conformance with ISO/IEC 9126; 13. The structure of GQM process will be complex if the difference in the quality focus emerges among stakeholders. This will require further iterations of GQM which will be costly for an organization (especially small or medium organization); 14. The model based on GQM is difficult to maintain;

1.4.9 Quality models in software packages methodology

1.4.9.1 Description of the method

This method has been proposed (Carvallo et al., 2002a and 2002b and 2003) to deal with requirements definition and decomposition. Requirements addressed by this method are quality requirements and the model used is the ISO/IEC 9126 quality model.

1.4.9.2 Activities of the method

Quality requirements are described in a structured methodology which is organized in the following steps:

- Defining the domain: examine and describe the domain of interest with the collaboration of experts;
- Determining quality characteristics;
- Defining a hierarchy of sub characteristics;
- Decomposing sub characteristics into attributes ;
- Decomposing derived attributes into basic attributes;
- Stating relationships between quality entities to determine the complete quality model. Various types of relationships can be identified: collaboration, damage and dependency;
- Determining measures for attributes: select measures for all attributes (basic) and derived context-free attributes;
- Collecting feedback to refine and extend the requirements.

1.4.9.3 Analysis and discussion of the method

This method presents the following advantages quoted by authors (Carvallo et al., 2003):

- Well structured and gives a detailed description of software product quality requirements definition activities;
- Easy to compare quality requirements with package selection descriptions;

- The quality models obtained with this methodology can be supported by packages selection tools;
- The methodology increases reusability.

Nevertheless, the following drawbacks are noted:

- Restricted to software package selection domains;
- Does not indicate how to identify, specify, decompose and control quality requirements in the proposed steps;
- Focused on external attributes because package suppliers do not give access to the package code;
- Not focused on the mapping activities of quality engineering instruments with the product definition phase of the life cycle.

1.4.10 Quality specification strategies for embedded software

1.4.10.1 Description of the method

The proposed method is a "Multi party chain" strategy which deals with software quality of embedded software (Solingen et al., 1999b). It was developed by the Spirits project and is based on user's perceptions (different stakeholders) of software product quality requirements. Stakeholders involved in the product usage should have responsibility to define quality requirements of the product. As these stakeholders (buyers, users, developers and project manager) have different views of the software product, Solingen points out the importance of supporting communication between these parties about product quality. Availability of these quality requirements facilitates translation of different interpretations and negotiation of these requirements among the involved parties. Requirements addressed by this method are quality characteristics of the software product. The quality model used is the ISO/IEC 9126 standard. Tools/techniques supporting this method are interviews and UML model.

1.4.10.2 Activities of the method

The method is structured as follows:

- Identify all the involved parties (users and stakeholders) in definition of software product quality;
- Use a model (multi-chain) to capture quality requirements of the relevant parties and trace them easily;
- Make a series of structured interviews with representatives of all parties in order to obtain a complete view of the quality requirements formulated in standardized quality terms (ISO/IEC 9126);
- Produce a consensus on the relevant quality characteristics of the software product.

1.4.10.3 Analysis and discussion of the method

The strengths of the method (Solingen et al., 1999b) are:

- Quality view of the user: users and stakeholders involved in the software product project participate in defining quality requirements;
- The multi party chain model : allows one to capture quality requirements of the involved stakeholders and users;
- Communication among the involved parties about product quality: facilitates resolution of conflicts between involved parties and helps to build a consensus about software product quality characteristics.

However some drawbacks are identified:

- There is no mention of how quality requirements are captured in the “multi chain” model, specified and retraced;
- How do conflicts between parties get resolved and are there any comprehensible guidelines to provide the consensus view on quality requirements and their relationships?

- Not focused on the mapping activities of quality engineering instruments with the product definition phase at the early requirements stage;
- Inability to reuse quality experiences in other projects and companies.

1.4.11 Method SHEL (Software and HardwarE and Live ware)

1.4.11.1 Description of the method

The approach proposed by Felici (Felici et al., 2000) deals with the integration of different types of requirements, which are defined over software, hardware and live ware (human) resources. Requirements defined by this approach are: cognitive, functional and quality. The SHEL model supports a systemic view which in turn supports the definition of different types of requirements related to system (software and hardware) and human aspects (human roles, interaction, and help in breakdown-situations).

1.4.11.2 Activities of this method

The definition process of these requirements is composed of 6 phases (Figure 1.37):

- **Work analysis:** is a profound analysis of the work system of the specific environment.
  - Studying the way in which the productive process is performed taking into account all the resources that contribute and interact in the process execution;
  - Producing models and processes as tools supporting process performance, objects in the work process, interactions, social and work practices in order to describe the existing work system with its critical issues and weaknesses.
- **Identification of user needs and critical issues:** elicit critical issues due to the knowledge distribution among the SHEL resources and their interaction.
  - Providing a basis for alternative design considerations represented by various prototypes and design models early in the design process;
  - Ascertaining suitable knowledge distributions for an effective use of the resources.

- **Definition of SHEL requirements and Design of the SHEL system:** All the collected information contributes to defining the requirements and architecture of the system according to the SHEL model.
- **Definition of functional, cognitive and quality requirements**: Quality requirements are defined according to the UCD approach (User Centered Design) (Felici et al., 1998) which is based on the user viewpoint to define quality needs for the system. A quality model is defined with quality characteristics by using the task analysis technique. The tool (SQUID) (Felici et al., 1998) and (Kitchenham et al., 1997) for data acquisition is used to control and evaluate software quality.
- **Design of prototypes and Mock-ups:** requirements are mapped into "design patterns" represented in prototypes, mock-ups, design models and scenarios.
- **Validation by experts in the work environment:** The last phase of one iteration cycle in SHEL oriented requirements engineering approach is the validation by the domain experts:
  - System compliance with requirements is evaluated. The evaluation takes into account "measurable criteria" such as performance and criteria such as usability, cognitive workload and level of cognitive support;
  - Requirements and the projected system are validated in the real system environment.

Techniques used in this method are: observations, interviews, heuristic analysis, video recording and checklists for capturing information on the productive process. Prototypes, design models, patterns, mock-ups and scenarios are used for validating the work environment.

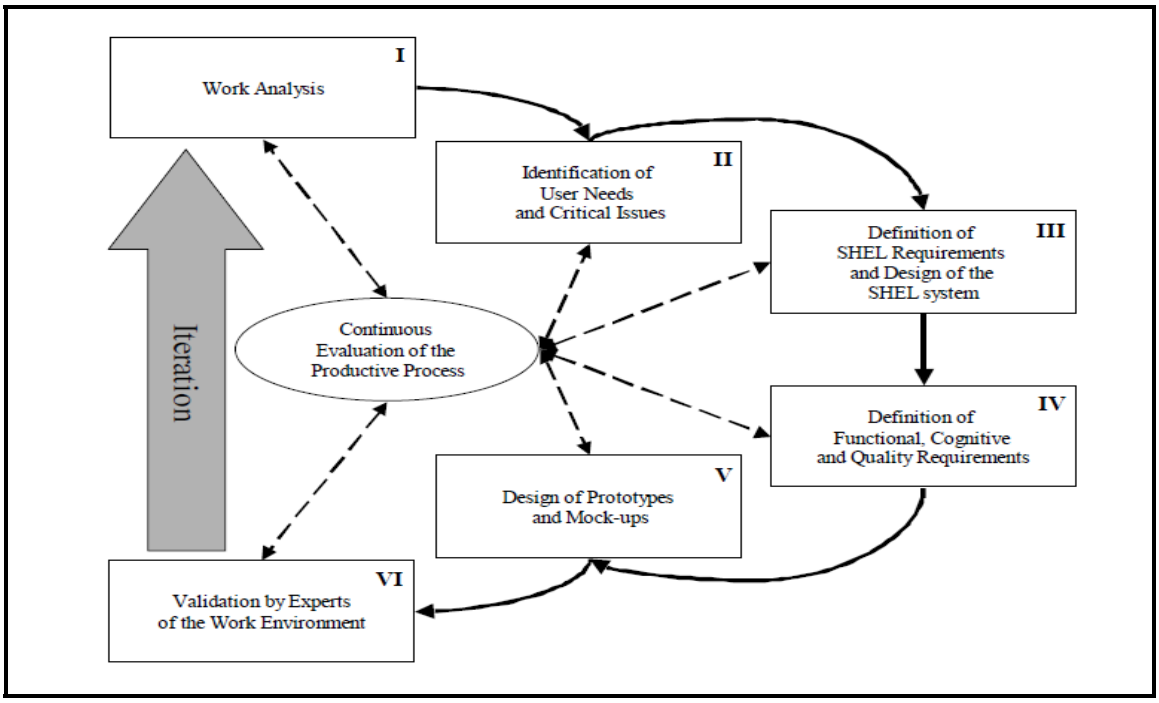


Figure 1.37 Phases of the process for defining requirements
Extracted from Felici et al., (2000)

1.4.11.3 Analysis and discussion of the method

This method deals with the integration of different types of requirements defined for software, hardware and live ware resources. It defines requirements by a systemic requirements engineering process and represents quality requirements as user needs which have been identified in the first "work analysis" phase of the SHEL oriented process. The third phase, "Definition of SHEL requirements and Design of the SHEL system", defines the requirements and architecture of the system according to the SHEL model which represents the starting point for defining quality requirements and other requirements as stated before (cognitive and functional). This method seems to be costly and does not support all features required for an integrated method and does not indicate how to identify quality requirements from SHEL system and design architecture (Djouab and Suryn, 2006). In fact, there is no mention of any technique for extracting QRs from SHEL requirements and design of the SHEL system. Finally, this method is not focused on the mapping activities of quality engineering instruments at the early requirements stage.

1.4.12 BMM (Business Motivation Model)

1.4.12.1 BMM definition

BMM is an intentional model which focuses on intentions, motivations and reasons, and deals with complex human and organizational issues (BRG, 2007). BMM has been introduced in the literature review because business goals are important drivers of the system as described by MOQARE and ATAM methods and will be part of the research solution.

As claimed by Business Rules Group (BRG, 2007), the BMM is designed to provide a structure for developing, managing and communicating business plans in an organized manner. BMM has been proposed as a standard under the Object Management Group. It is a simple and compact standard that provides a metamodel for enterprise-specific motivation models. BMM contains and organizes the elements of its business governance: vision and mission, influences and assessments, goals and objectives, strategies and tactics, as well as business policies. It references other relevant elements of its business models (its business processes, business rules, organization units, assets, resources, products, services) that are contained in related models built using specifications outside the BMM scope. Models are expressed in a Unified Modeling Language (UML) standard (Figure 1.39 for detailed metamodels). Table 1.10 summarizes the definitions of the core concepts of BMM.

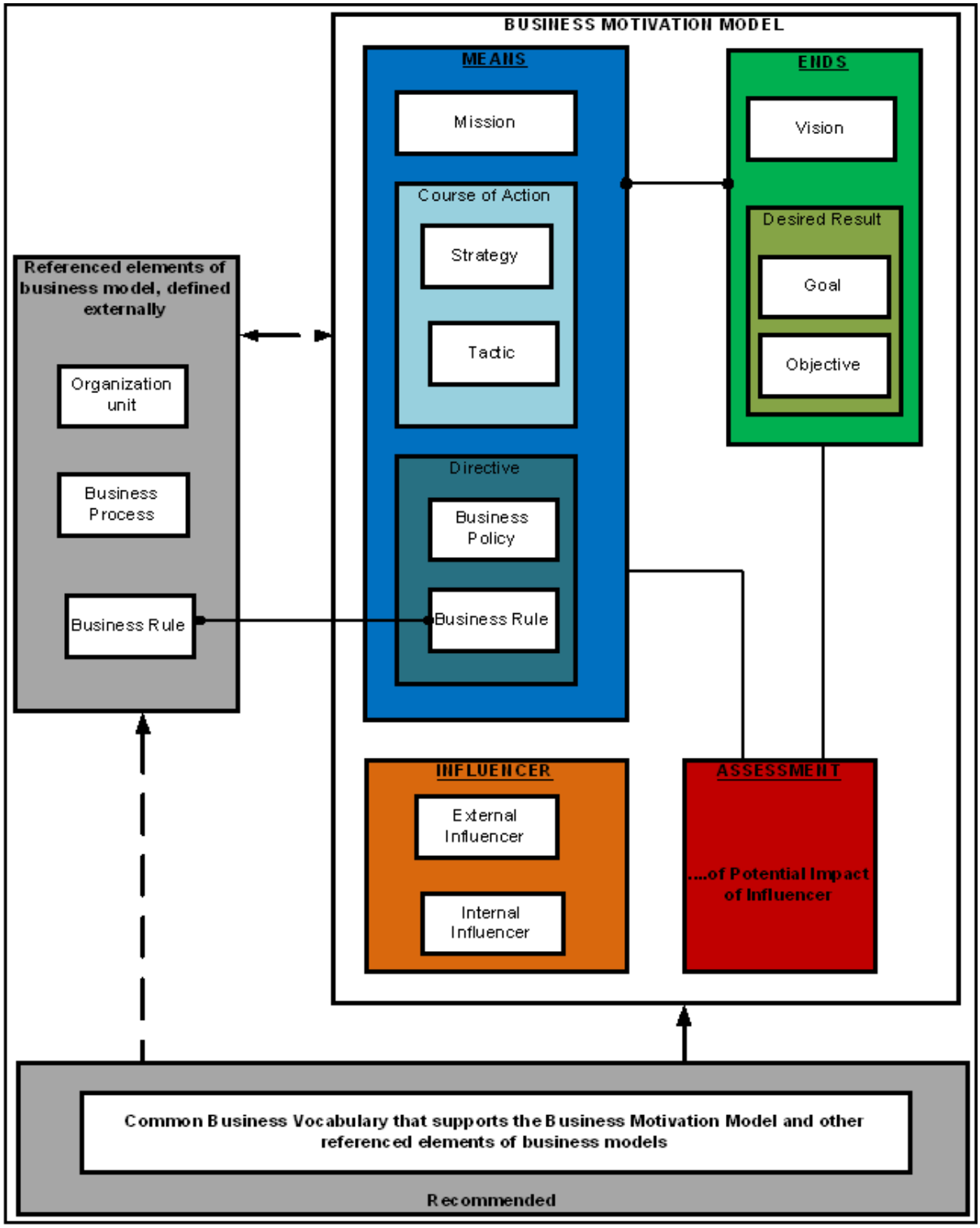


Figure 1.38 Business Motivation Model Framework
Extracted from Deng (2006, p.35)

Table 1.10 BMM concepts descriptions

Artifacts	Description (from (Deng, 2006))	Commentary
Ends	What an enterprise wants to be. Examples: 1. Develop new lines of business 2. Moving into new markets 3. Maintain its current position in the market	Ends do not say how the goals will be achieved
1. Vision	1. An overall image of what the organization wants to be or become	
2. Desired results a. Goals **b.** Objectives	2. Are more specific a. Tend to be long term and defined qualitatively b. A step along the way towards a goal and is quantitative.	
Means	What the organization needs to achieve what it wants. It indicates capabilities that can be exploited to achieve the desired results	Means do not indicate business process necessary to exploit them and responsibility for such tasks
1. Mission	It indicates the ongoing operational activity of the enterprise. It covers all strategies and complete area of operations	
2. Course of Action **a. Strategy** **b. Tactic**	What the enterprise has decided to do and what has to be done a. Strategy tends to be long term and broad in scope. It is implemented by tactic b. Tactics tend to be short term and narrow in scope. Tactic may contribute to implementation of more than one strategy.	1. Course of Action does not define how well it has to be done a. Strategies are selected to move the enterprise towards its goals b. Tactics are selected to ensure that it meets its objectives
3. Directive **a. Business Policy** **b. Business Rule**	a. Business policy governs, controls, guides and shapes strategies and tactics. It defines what can be done and what must not be done. It sets limits on how it should be done b. Derived from business policy, it needs to be defined and managed for consistency and completeness. It provides specific solution when a course of action fails and specific resolutions to conflicts arising among ends.	**1.** Every directive must be explicit and recorded in an official manner **2.** All courses of actions must be governed by some directive
Influencer	Any changes affecting the enterprise in employment of its Means or in achievement of its Ends	
1. Internal	1. Internal changes are: a. Infrastructure b. Issues c. Resource habit d. assumptions	
2. External	1. External changes are: a. Environment; Technology; Regulation b. Supplier c. Customer d. Competitor and partner	
Assessment	Judgment about the influence of an Influencer on the ability of the enterprise to achieve its Ends or use its Means	
1. Potential Impact **a. Risks** **b. Potential reward**	1. Identified to support assessments	

1.4.13 Synthesis of described methods

The present section discusses and analyzes the four classes of software QRs management methods described in the previous section.

The first class: *Business goals oriented quality methods* (Space-Ufo, MOQARE and ATAM) uses business goals as main forces in the quality requirements management process. Space-Ufo identifies quality needs according to the characteristics of the business system. MOQARE is applicable to QRs derived from business goals. ATAM supports evaluation of given architectural alternatives with respect to quality requirements attributes. However, there is:

1. No mention of how to describe and model business characteristics in a structured way to identify quality needs (Space UFo);
2. A lack of a systematic way to write scenarios (ATAM);
3. Focus on eliciting architecture-centered quality attributes (ATAM);
4. Absence of documentation of conflicts between quality concepts (MOQARE);
5. No direct integration of NFRs and FRs (MOQARE);
6. No support for non-technical stakeholders and novices (MOQARE).

The second class: *Aspect oriented quality methods* (*FDAF* and *Requirements model for quality attributes*) is based on the "Aspect" concept to describe QRs. The FDAF framework has been designed for a specific quality attribute at the architectural level. It has been developed to create architecture designs with NFRs aspects that cannot be described in the real time version of UML. However, FDAF is not concerned with the identification of QAs at the requirement level. It uses limited analysis tools and modeling constructs to describe a component's behavior and connections. *Requirements model for quality attributes* process defines quality attributes as crosscutting concerns and specifies them in a template. But there is no indication of how to identify quality attributes from system and user requirements.

The third class: *Goal oriented quality methods* (*IESE NFR, soft goal notation and Prometheus*) is based on a "goal" concept to describe QRs. The IESE NFR method is a well

defined process for eliciting complete and measurable NFRs, except that quality attributes are not derived from business goals and the tailoring stage is not supported by context rich scenarios as is the case for MOQARE method. The soft goal notation method is applicable to all types of QRs, but focuses on the documentation and negotiation of QRs, and not on their elicitation from business goals. The Prometheus method gives a detailed process to build a quality model for a specific domain project, but it seems to be difficult to apply particularly in the step of construction of the BBN quality model. In fact, the BBN quality model is restricted to a maximum of three decomposition levels and initialization of the BBN network (filling the probability tables) requires more effort from experts.

The last category of quality methods (*Quality models in software packages*, *Quality specification strategies for embedded systems* and SHEL (**S**oftware and **H**ardwar**E** and **L**ive ware)) lack a systematic way to manage QRs. They are either restricted to software package selection domains or do not indicate how QRs of the involved stakeholders are captured, not does it define QRs in the scope of a systemic requirements engineering process.

Some of the potential drawbacks of QRs management methods will be addressed by the research solution (Figure 1.40). The QRs management methods are: "S*oft goal notation", ATAM, IESE NFR method, MOQARE* and "*Requirements model for quality attributes*". For instance, the drawbacks of MOQARE and Soft goal notation methods (no integration of NFRs with FRs and NFRs not elicited from business goals) will be addressed in the research solution by defining concepts dealing with business goals and integration of NFRs with FRs.

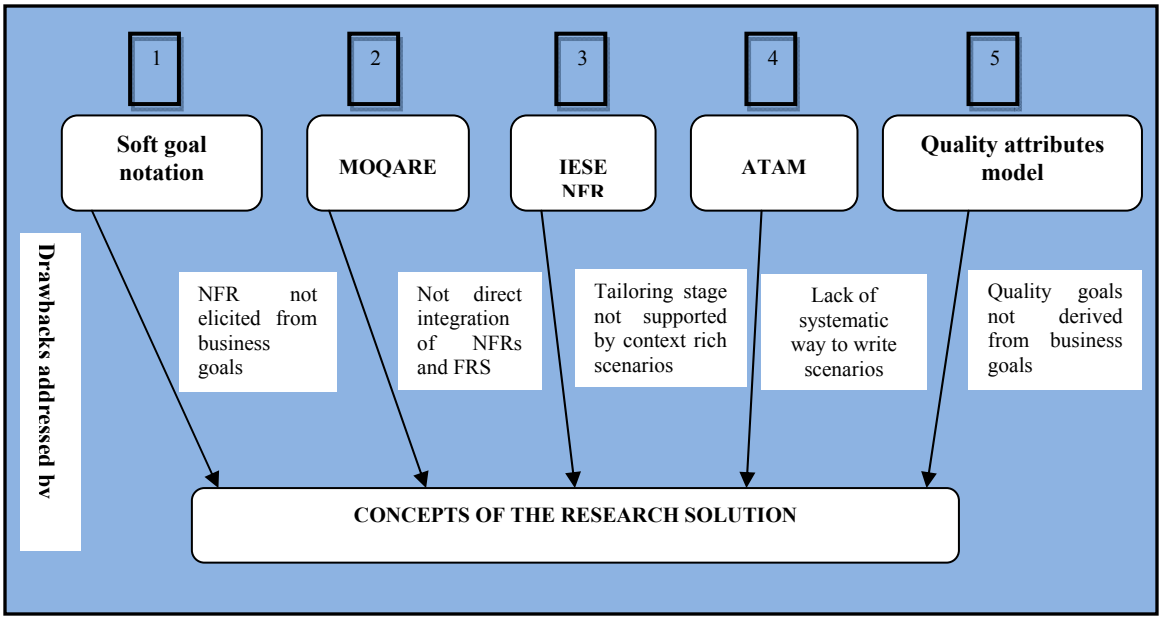


Figure 1.39 Drawbacks of the QRs management methods and the research solution

These QRs management methods are compared using criteria related to the management of QRs which are: identification, decomposition, conflict analysis, representation, documentation, derivation from business goals of quality attributes, consensus on quality definitions, quality standard and integration with FRs. These criteria have been chosen according to the identified drawbacks of the software QRs management methods.

Table 1.11 summarizes an assessment of the software QRs management methods according to established criteria. Table 1.12 establishes comparisons of the used artifacts of methods according to quoted criteria.

In summary, the described software QRs management methods presented some advantages and strengths which can be summarized in Tables 1.11 and 1.12:

- Decomposition and representation of quality requirements (as soft goal notation and IESE NFR methods);
- Conflict analysis among quality requirements (as IESE NFR and soft goal notation methods);

- Integration of quality requirements with functional requirements and architectural options (as IESE NFR and QAs model).

But in the most methods:

- The identification of QAs is partially covered and there is a lack of a structured way to show clearly how QAs are extracted from the original requirements (system/user requirements and business specifications);
- The conflict resolution among QAs is not addressed (except for ATAM, IESE NFR and soft goal notation methods);
- The derivation of QAs from the business specifications is not covered (except for ATAM and MOQARE methods);
- There is a lack of documentation of QAs and consensus on quality definitions;
- There is an absence of software quality engineering standards (except for IESE NFR method);
- The integration of QAs with FRs is not addressed (except for IESE NFR and QAs model).

Table 1.11 Summary of chosen methods
and their criteria assessment

Quality requirements methods	Characteristics and criteria								
	Identification of quality attributes	**Decomposition**	**Conflict analysis**	**Representation**	**Documentation**	**Consensus on quality definitions**	**Quality standard**	**Derivation from business goals**	**Integration with FRs**
MOQARE	Partially	Partially	No	Yes	No	No	No	Partially	No
IESE NFR	Partially	Yes	Yes	Yes	No	No	Yes	No	Yes
ATAM	Partially	Partially	Yes	Yes	No	No	No	Partially	No
SOFT GOAL NOTATION	Partially	Yes	Yes	yes	No	No	No	No	No
Quality Attributes Model	Partially	Partially	No	No	No	No	No	No	Yes

Yes: concept is well defined.

No: concept is not defined.

Partially: concept is mentioned but not defined.

Table 1.12 Comparisons of chosen methods

QRs methods	Characteristics and criteria									
	Identification of QAs	Decomposition	Definition	Conflict analysis	Representation	Documentation	Consensus on quality definitions	Quality standard	Derivation from business goals	Integration with FRs
MOQARE	Interviews	Checklist	Quality goals		Misuse tree				Checklist	
IESE NFR	Prioritized questionnaire	Quality model	QAs	Dependency analysis	Quality model			ISO/IEC 9126		Consolidation step
ATAM	Scenarios	QA characterization	QAs	Brainstorming session	Utility tree				Questions	
SOFT GOAL NOTATION	Interviews	Soft goal graph	NFRs	Catalogues of methods	Goal graph structure					
Quality Attributes Model	Interviews	Soft goal notation	QAs			Template				Aspect Oriented Method

1.5 Chapter summary

In this presented literature review, we investigated the main aspects of software QRs like existing QRs definitions and terminology used to specify them, important software quality engineering standards and QRs management methods developed during the last 2 decades.

In the section related to the QRs definitions, Azuma defines relationships between needs and requirements as "*stakeholder's needs (stated and implied) are collected and identified, then selected and specified to be transformed in QRs"*. Further, requirements elicited from stakeholders' needs are defined in 3 views of software QRs: quality in use requirements, external and internal quality requirements. In addition, SWEBOK defines NFRs as constraints or quality requirements. On the other hand, several authors including Suryn, Pfleeger, Lauesen and Hans Van Vliet highlight the importance of dealing with QRs at early stages and the difficulty to specify and verify them (Table 1.2). They also put emphasis on their modeling and representation. So, new methods and standards have emerged for this purpose.

The software QRs management methods section can be classified into three main categories:

1. The business oriented quality methods are based on the elicitation of business goals and business characteristics in order to define QAs of the software product. Example of such methods: ATAM, MOQARE and Space Ufo. ATAM and Space Ufo need to be supported by more structured QRs management techniques;
2. The aspect oriented methods (Quality model for QAs and FDAF) promote use of aspects to specify QAs that often scatter functional requirements. These methods are faced with the problem of applying aspects at the requirement level due to the strong interdependencies among NFRs;
3. The goal oriented methods (ISESE NFR, NFR framework and Prometheus) use the goal as the main guiding concept in QRs specification, refinement and conflicts resolution.

For the described software quality engineering standards section, four software quality engineering standards have been presented: the McCall quality model, the Boehm model, Dromey's quality model and ISO/IEC 9126: Software Product Evaluation: Quality Characteristics and Guidelines for their Use-standard (ISO/IEC 9126, 2004). ISO/IEC 9126 was part of the first generation of software quality engineering standards. It was improved by ISO/IEC SC7 WG6 experts to build the new standard for quality requirements specifications ISO/IEC SQuaRE 25000. ISO/IEC SQuaRE 25030 is the standard enabling software product quality requirements to be specified, tracked, validated and evaluated from different perspectives (acquirer, developer and evaluator) (section 1.2).

Chapter 3 will introduce the methodological aspects of the research that lead to the research goal and objectives and the main research steps used to design the proposed method SOQUAREM.

CHAPTER 2

RESEARCH OBJECTIVES AND METHODOLOGY

This chapter describes methodological aspects of the research project. Section 1 presents research issues and fundamental questions related to the research project. Section 2 describes the main goal and research objectives. Section 3 describes in detail the required steps to accomplish these research objectives. Last section concludes the chapter.

2.1 Introduction

Research issues identified from analysis of literature review refer to limitations of engineering approaches in addressing quality requirements. The majority of the described methods in chapter 1 such as MOQARE, IESE NFR, Soft goal notation, ATAM and "Quality attributes model" deal partially or not at all with criteria related to: identification, decomposition, representation, conflict analysis and documentation of QAs (Tables 1.11 and 1.12 in section 1.4) .

This research project addresses the limitations related to the identification, conflict analysis, representation, documentation, derivation from business goals of quality attributes, quality standard and integration with FRs and proposes the design of a quality requirements engineering method and its model.

The method should support convenient refinement techniques and linkage mechanisms by which QAs are obtained from the stakeholder's business goals. The linkage mechanism is supported by the quality standard ISI/IEC 25030 to infer the right QAs. Secondly, the method should provide efficient ways to support representation, documentation and integration of QAs with the FRs model.

The method will apply a dedicated process of managing quality attributes (Figure 2.1). The process will help the person responsible for defining new software product quality attributes

to identify and refine business goals, link them to QAs, represent them in a personalized quality model, specify them in a template and finally integrate them into the FRs model.

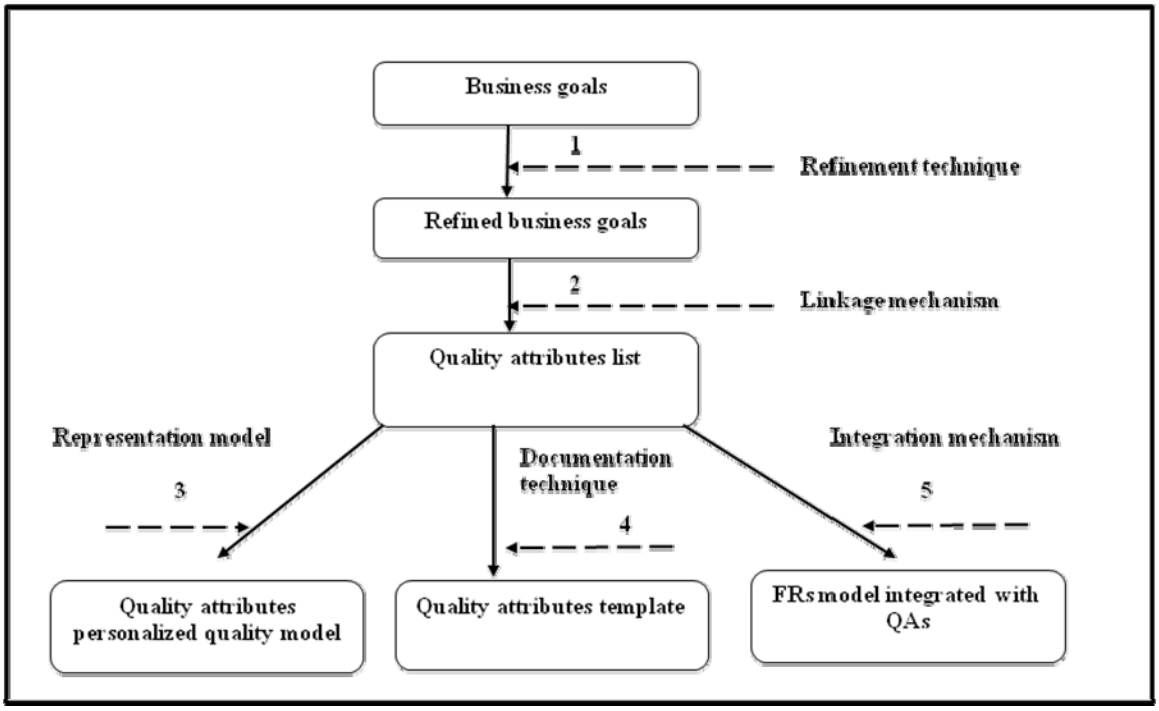


Figure 2.1 Process of managing quality attributes

The fundamental questions related to the research project are organized around five basic elements:

- Derive the stakeholder's business goals into refined business goals (question 1) ;
- Link the refined business goals into the corresponding quality attributes according to ISO/IEC SQuaRE 25030 quality standard (question 2);
- Integrate quality requirements into the personalized quality model and retrace them to their original requirements (question 3);
- Specify and document quality requirements (question 4);
- Integrate the QAs with the FRs model (question 5).

2.2 Research Goal and Objectives

The main goal of this research is:

"To support the software product definition phase with a systematic management method of quality requirements."

To pursue to this goal, the research objectives are:

a. Develop a structured quality requirements engineering method: **SO**ftware Product **QUA**lity **R**equirements **E**ngineering **M**ethod (SOQUAREM) supported by the quality standard ISO/IEC SQuaRE 25030.

Sub objectives are:

- Development of an identification technique of quality requirements;
- Development of a representation model of quality requirements;
- Development of a documentation formalism of quality requirements;
- Development of an integration technique of quality requirements with the FRs model;

b. Develop the process model representing concepts and phases of SOQUAREM method.

2.3 Research Methodology

The research methodology designed to attain the research objectives includes the following research steps (Figures. 2.2 and 2.3):

a) Exploration phase: this step studies the main concepts and definitions related to the software QRs management domain**.** It includes a literature review of:

- Software QRs definitions and concepts (details in chapter 1);
- Software quality engineering standards ISO/IEC 9126 and ISO/IEC 25030 (details in chapter 1);
- Quality requirements management methods (details in chapter 1);

The literature review revealed the following:

Software quality requirements:

- Easy to specify but difficult to identify, test and control;
- Difficult to define in the same terminology when stated by different stakeholders;
- Often conflicting among each other which is difficult to resolve;
- Often scattered and tangled with functional requirements.

The Quality standard ISO/IEC SQuaRE 25030 will be used in the research project to support the proposed software QRs management process.

The QRs management methods deal partially or not at all with identification, documentation, conflict analysis and integration with the FRs process.

b) Analysis phase: consists of analyzing the existing software QRs management methods (chosen from literature review) in their cases studies to know to what extent they address management of software QRs. Further, a questionnaire is developed and distributed in the industrial circle and the collected data is analyzed to determine the current state of the software QRs engineering practices in industry. Finally, the obtained data from industry and academia are analyzed to define the future requirements of the research solution. The analysis phase is divided in three sub phases (details in chapter 3):

i. **Analysis of existing QRs management methods:** the goal of this phase is to determine the strengths and weaknesses related to applicability of the recognized methods addressing quality requirements in the scientific environment and industry (Methods are: MOQARE, IESE NFR, Soft Goal Notation, ATAM and FDAF).
   The research focuses on analyzing the existing and known implementations of each method by considering applicability of the method (the case study) and elements used in the engineering process of the method as: a) identification of quality requirements-related activities; b) identification of used techniques/tools (questionnaires, checklists, templates

and patterns) ; c) identification of implied actors; d) identification of the used quality model and standards and e) identification of results and artifacts produced by this method. The approach adopted during this analysis describes the applicability of methods by analyzing their case studies in the applicative domains and identifying their strong and weak points. The results envisaged are indicators describing the strengths and weaknesses of the applied methods in industrial and scientific communities (details in chapter 3.1).

ii. **Quality requirements data collection practices in industry**: the goal of this phase is to determine the current state of the quality requirements engineering practices in industry. A questionnaire is distributed in industry and its sections are defined as follows:

- Information on the respondent;
- Companies and stakeholders;
- Processes;
- Methods;
- Software quality engineering standards used in the applicative domain.

Results envisaged are indications about the development of the software QRs engineering practices that could be proposed to industry (details in chapter 3.2).

iii. **Analysis of resulted indicators from industry and academia environments**: the goal of this phase is to analyze industrial and academic indicators obtained in the two preceding sub phases and to identify critical needs seen by industry in the field of software QRs management. Important conclusions and justifications of the proposed solution will be formulated. Analysis is carried out in the following categories (details in chapter 3.3):

- Identification of software QRs;
- Representation of software QRs;
- Documentation of software QRs;
- Integration of software QRs with the FRs model;
- Quality standard used.

c) **Design of the engineering method (SOQUAREM) and the associated process model**: it includes the creation of a quality engineering method and the associated model addressing the identified needs of the industrial software development environment. The design phase is divided into two subsequent phases:

i. **Development of the main concepts of SOQUAREM:** details the different concepts involved in the software QRs engineering process (BMM and BCT, scenarios template, utility tree and QAs template) (details in chapter 4.1).

ii. **Development of the process model and the SOQUAREM method:** describes the main phases of the SOQUAREM process (details in chapter 4.2). Each phase is described with three parts: input and output artifacts and used techniques and standards. The phases are:

1. State the business goals;
2. Refine the business goals;
3. Link the refined business goals to quality attributes;
4. Build quality attributes scenarios;
5. Consolidate quality attributes;
6. Link quality attributes to the functional process.

d) **Application of the method:** This step applies the method in an illustrative example and evaluates to what extent this method addresses software QRs management techniques (identification, representation, conflicts resolution and documentation…). The application phase is divided into two subsequent phases:

i. **Development of the exploratory case of SOQUAREM:** build an illustrative example for building an automation system to clarify the core ideas of SOQUAREM method and its practical relevance to the software product definition phase (details in chapter 5). Each phase of the SOQUAREM process is applied in the example.

ii. **Analysis and evaluation of the method:** the goal of this sub phase is to identify constraints and corrective measures in order to improve the method and define the research avenues (details in chapter 5). Analysis and evaluation are carried out in industry and academia (Workshop session) and on the ISO/IEC SC7 System and Software Engineering committee level. The adopted method to realize this evaluation is explained in the following points:

1. **Evaluation of the developed method**
   - By international experts in the software quality field;
   - During organized workshop session.

2. **Presentation of the method (once published) on the committee level of ISO/IEC SC7 - System and Software Engineering**
   - Direct co-operation with experts of the working group SC7 WG6 - software quality measurement and evaluation-.

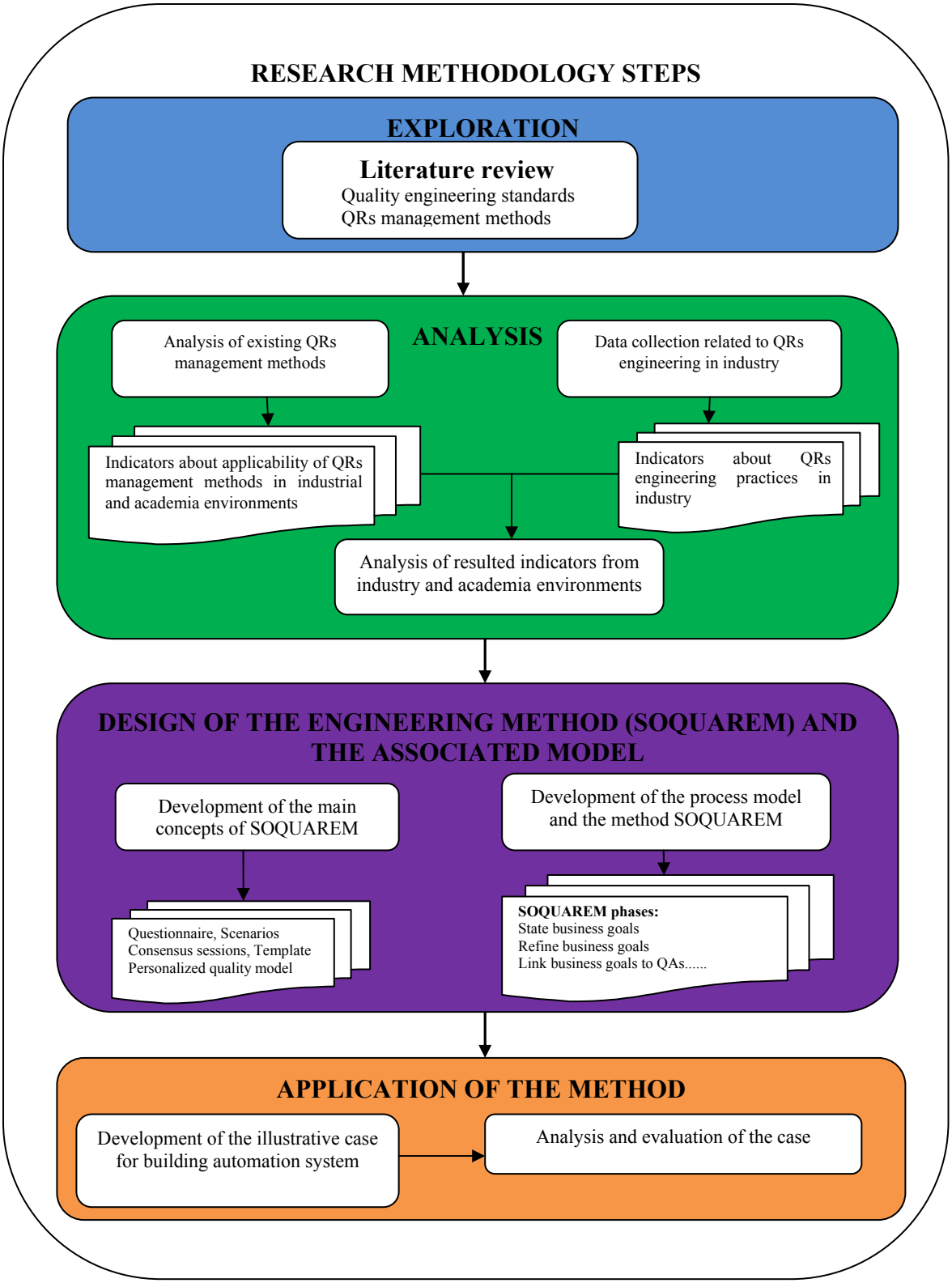


Figure 2.2 Research Methodology

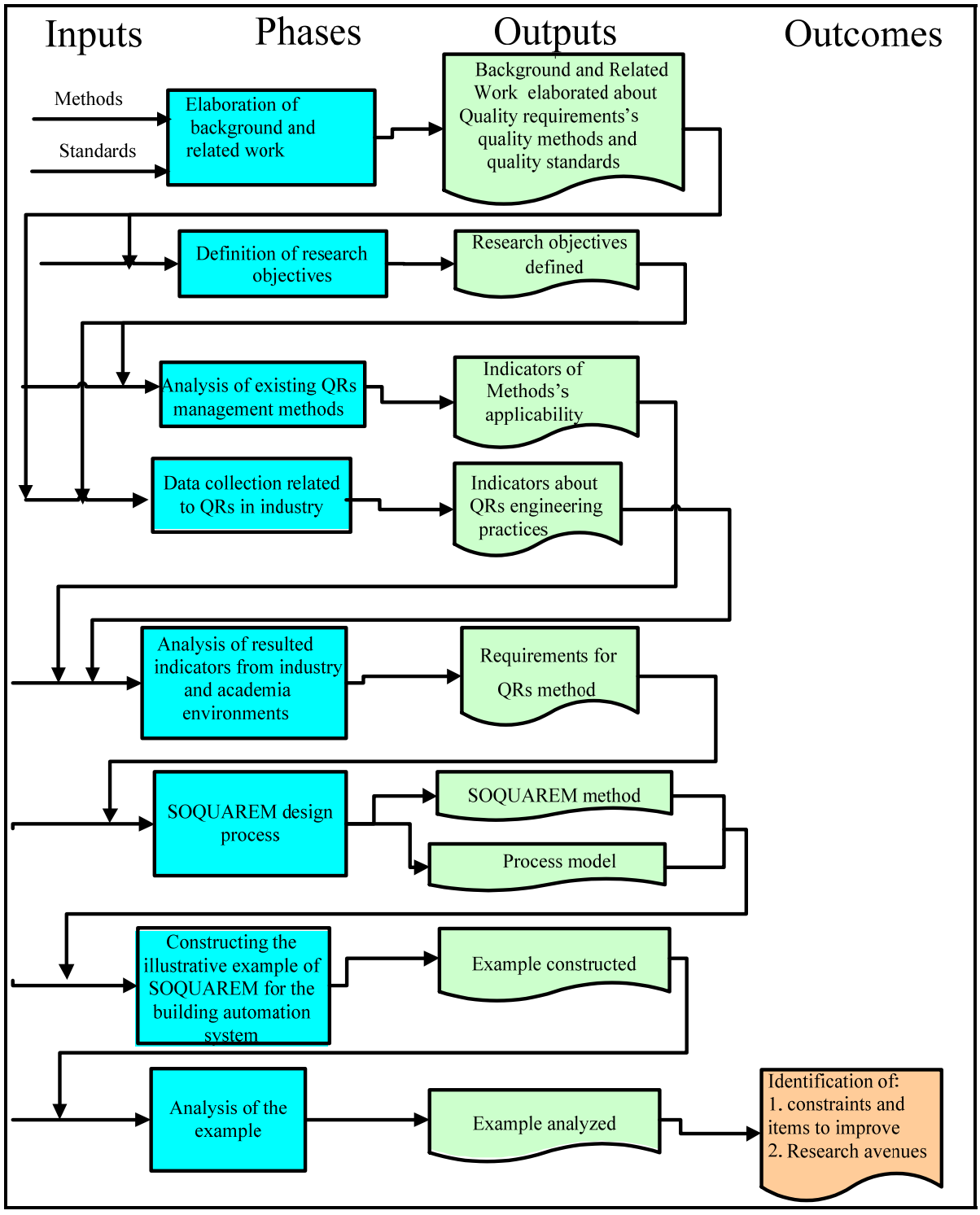


Figure 2.3 Research Methodology

2.4 Chapter summary

This chapter presented research methodology which addresses the research project of systematically identifying, specifying and representing quality requirements in processes and models for the software product definition phase. Fundamental questions related to the research project have been presented followed by the research objectives. Research steps to attain the stated objectives have been described in a dedicated-four-phase analysis methodology and are: exploration, analysis, design of the software quality engineering method (SOQUAREM) and the associated process model and finally application phase for the method. The exploratory phase is related to the literature review studying concepts dealing with the quality requirements such as standards and methods. This phase provides the current state of the art quality requirements subject. The analysis phase provides, in one part, indicators about the strengths and weaknesses of methods applied in the industrial and scientific communities, and on the other part, indicators about development of the quality requirements practices that could be proposed in industry. Analysis results are used to justify the future proposed quality engineering method and define the requirements for its design. The two last phases are related to the design and application of the software quality engineering method (SOQUAREM). The design phase describes concepts of the method and its process model. The application phase develops an illustrative example describing the application of the designed method and evaluates it in industrial and academic environments.

The next chapter studies in detail the analysis phase of the research methodology.

CHAPTER 3

RESEARCH EXECUTION

Chapter 3 describes the details of the research execution (Figure 3.1). Its main purpose is to justify the design of the QRs engineering method and define the requirements for this design. Section 1 discusses and analyzes applicability of existing QRs management methods in their respective case studies by establishing their strengths and weaknesses (Table 3.2). Methods are also assessed according to established QRs management criteria and compared to their used artifacts in case studies. Section 2 presents an overview of the the current situation of quality requirements environment in the industrial circle where a questionnaire is used and the collected data is analyzed. The analysis of resulted indicators from applicability of QRs management methods in industrial and academic environements and from QRs engineering practices in industry is also presented in this section. The critical needs are identified from the domain representatives in industry and relevant conclusions and observations are stated. From these conclusions, future requirements of the proposed research solution are formulated in the third section. An overview of the proposed method, justifications and added values are pinpointed. Section 4 concludes the chapter.

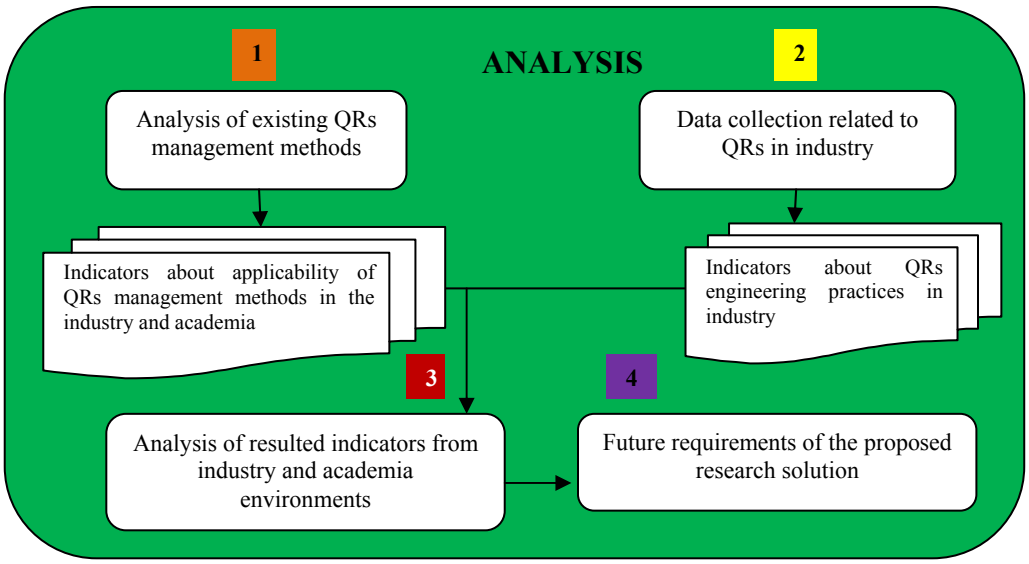


Figure 3.1 Research execution

3.1 How to apply methods for quality requirements management

To study and discuss the applicability of existing QRs management methods, the following case studies from the literature review were chosen based on their availability. They are:

1. FDAF (Formal Design and Analysis Framework): use the aspect concept and formal methods to design and analyse NFRs (Dai et al., 2005 and 2006) and (Cooper et al., 2004).
2. MOQARE (Misuse Oriented QuAlity Requirements Engineering): use business goals and misuse concept to describe quality attributes (Hermann et al., 2007a and 2007b);
3. ATAM (Architecture Tradeoffs Analysis Method): use business goals and scenarios to describe quality attributes (Kazman et al., 2000); (Jones, 2001); (Gallagher, 2000); (Bass et al., 2003); (Boucké et al., 2006) and (Venckeleer, 2006).
4. IESE NFR/ASPIRE[2] (Analysis of Software Product In Requirement Engineering) (Doerr et al., 2005);
5. Soft goal notation (or NFR Chung framework): use goals as a driving force to elicit and refine NFRS and to guide the design process (Chung et al., 1994 and 1995).

Table 3.1 describes the chosen QRs management methods with their main concepts and designed levels

[2] IESE NFR is the same method as ASPIRE

Table 3.1 QRs management methods with their concepts and designed levels

Concepts / QRs management methods		Paradigms and concepts					
		Aspect Oriented Paradigm (AOP)	Business Oriented Paradigm (BOP)	Goal Oriented Paradigm (GOP)	MisUse Case Concept	Scenarios concept	Formal concept
MOQARE	Requirement		*		*		
	Architectural						
IESE NFR	Requirement			*			
	Architectural			*			
ATAM	Requirement						
	Architectural		*			*	
SOFT [3]GOAL NOTATION	Requirement			*			
	Architectural			*			
FDAF	Requirement						
	Architectural	*					*

3.1.1 Analysis and discussion of applicability of QRs management methods

The present section analyzes and discusses the applicability of QRs management methods according to their case studies.

The FDAF aspect oriented approach has been applied in three case studies: building security for online banking, achieving a performance response time for the ATM banking system and

[3] Soft Goal Notation is also called the NFR Chung framework

analyzing the resource utilization performance aspect for the domain name server system (DNS).

1. The first case study illustrated building a Role Based Access Control (RBAC) (Dai et al., 2006), a design of an aspect of architecture for an online banking system using the FDAF framework. RBAC aspect is adapted from well established RBAC security patterns. The FDAF framework has been used to define the role based access control (RBAC) aspect on the basis of the security pattern and model in a UML architecture design. The RBAC model has been translated into an Alloy specification and analyzed. The analysis results help architects to detect inconsistencies in the multiple systems' RBAC policies early in the design. A parallelogram notation is used to present aspect information and is incorporated into the standard UML to indicate where in the static or dynamic model, all or in part of the aspect needs to be included. The advantage of translation approaches is that verification and validation techniques and tools can be applied to the source semi-formal notation as UML. This case study has shown that the definition of the RBAC security aspect is adequate and security aspects could be reusable with certain assumptions and customizations. The definition of the RBAC aspect is refined with a new attribute called "assumption" which describes possible assumptions about the system making this aspect easily applicable. Therefore, building the RBAC security aspect into the software architecture helps to meet the enterprise level security requirements.

2. In the second case study (Dai et al., 2005), the response time performance aspect has been modeled in the UML architecture design by using the stereotype PAstep. Rapide's analysis tool supports architects with detailed analysis of the system's behaviour simulation. Results of the response time analysis are available in the early design. However, Rapide's analysis tool is limited by its capacity to provide analysis results (in the graphical browser) when the number of simulated events increases.

3. In the third case study (Cooper et al., 2004), the problem of overloaded component has been resolved by defining the "Resource utilization" aspect with a set of UML

stereotypes. Armani's analysis tool in FDAF was used to provide architects with detailed analysis information about which component is the bottleneck (overloaded and busy all the time). However, there are some limitations of Armani's analysis tool in this area: the mathematical assumptions restrict the systems they model. For example assumptions that all components are being executed sequentially are not applicable to systems where components are executed concurrently. Another concern of the Armani tool is that it does not calculate automatically the property "sOverloaded" and must be changed manually by architects.

In conclusion, the FDAF framework is an interesting approach to create architecture designs with NFRs aspects that cannot be described in the real time version of UML. The major contribution of FDAF is that it integrates the semi-formal UML with the formal methods into an aspect oriented framework. The aspect model is based on one specific aspect which makes it simpler than a traditional mixed model. Application of this method in the case studies showed that NFRs are a powerful tool to evaluate architecture designs and to predict early design errors and be able to improve them before delving into the implementation features (Djouab and Suryn, 2007b). However, it deals with one specific aspect composed of multiple sub aspects such as performance, response time and resource utilization and there is no mention in FDAF of how to deal with interdependencies between NFRs (aspects).

IESE NFR method has been applied in three industrial domains: wireless plant control system, multi-functional printer systems and geographical information system (Doerr et al., 2005).

1. In the wireless plant control case study, the prioritization of the quality attributes (QAs) (efficiency, reliability and maintainability) and tailoring of their associated quality models (QMs) are done in the first workshop in order to be available for the elicitation process. Efficiency requirements are elicited in the first workshop and reliability with maintainability requirements are elicited in the second workshop. The elicitation process was supported by the quality models, checklists and dependency analysis activity for

identifying and resolving conflicting requirements in the early phase. One observes that many NFRs missing before are now elicited. However, much time was spent during the tailoring process to resolve terminologies problems and improve the quality models.

2. For the second case study, two workshops were held: one for customizing the quality model (QM) and the other for the NFRs elicitation. Refinement of the QA selected in the multi-functional printer systems (efficiency) differed from the wireless system in the sense that a new need emerged: requirements management support for clarifying the NFRs. Especially for the embedded system (high integration of software and hardware), requirements management support is of great importance in order to palliate the difficulty defining all the requirements. Furthermore, the specified NFRs must be detailed in the subsequent development phases. Another aspect discovered in this case study is the interdependency of the functional requirements with non functional requirements which results in additional effort through iterations.

3. In the geographical system, a particular QA was selected (security) with the associated quality model based on ISO/IEC 9126 and the security domain experts. The experience acquired in this case study, in particular during the elicitation process, was attaching metrics to this QA, the importance of integrating functional requirements and architectural options and a need for significant rework on the architectural level in order to integrate NFRs.

In summary, IESE NFR is project and domain dependent. QAs are influenced by the project–specific variations and elicited according to priority of the industrial application, type of project and quality viewpoints of the different workshop participants. In addition, IESE NFR is costly in time because the requirement management support is performed in iterations and the size of checklists will be large with the growth of conditions and alternative sections (Djouab and Suryn, 2007b). In fact, the experience based artifacts (models, checklists and templates) have to be maintained to be used efficiently. Furthermore, the application of IESE NFR depends on functional requirements and architectural options. Experience showed that

NFRs, functional requirements (FRs) and architectural options (AOs) must be intertwined because refining NFRs is not possible without detailing functionality or architecture. Major rework has to be done to integrate NFRs in architecture (case study 2).

The NFR Chung framework case study (Chung et al., 1994) is a good example to represent the relevant concepts and methods for dealing with NFRs during the software development process. The NFR-Assistant tool defined two NFRs catalogues "Security" and "Performance" with their associated techniques. However, there is a need for definition and use of more specialized methods requiring additional domain expertise. There is also a need for use of the framework by a variety of users dealing with a variety of non-functional requirements (not limited to accuracy, security and performance), a variety of domains and a variety of system characteristics. This case study showed that capturing domain expertise early in the process and participation of stakeholders in resolving quality terminology issues are important steps in the framework. In addition, it has been mentioned that training a variety of users (developers and administrators) in the use of the framework by a (cost-) effective means is required. The NFR assistant tool should be extended by a larger set of goals and methods to see if it could be accommodated and graphically represented.

For the MOQARE method (Hermann et al., 2007a and 2007b), the requirements elicitation was guided by the four steps of the process, the misuse tree and checklists. The misuse tree gives an overview of the requirements and is used to structure interviews and support the iterative requirements elicitation process. In fact, for each iteration, a branch is created to support interviews bringing new results. However, MOQARE requires a method specialist to represent the stakeholders' requirements into a misuse tree. The produced misuse tree in this case study contained two iterations. On the first level, there were two quality goals, 10 threats and 35 countermeasures (13 were quality goals and three of these countermeasures were analyzed further, leading to 10 more threats and 15 countermeasures). The 15 quality goals belonged to all six categories of ISO 9126. In addition, one observes on the first level of the analysis that only mere data was important in the case study, but later on the MOQARE

analysis showed that the whole process of data input, processing and output had to be controlled.

In summary, MOQARE is an emerging method supporting systematic identification of QRs from business quality goals. However, the method seems to be more complex and difficult to be understood by non technical stakeholders. In addition, it has been mentioned that main quality issues captured by MOQARE are not measurable at an early stage and conflicts between quality attributes are not documented. MOQARE needs to be validated in a real context with a large spectrum of users.

During the application of ATAM to a large government-sponsored simulation system (the Wargame 2000 system a highly complex real-time simulation system), the results of this evaluation reported some benefits (Jones, 2001) like: “The stated goals of the ATAM evaluation were met” and “The evaluation allowed a focus on the entire system rather than narrow or short-term concerns”. The case study shows that ATAM is appropriate for use when a system is in development and improves understanding of architectural issues for the future versions of the system and stakeholder communications.

The work of Gallagher describes the application of ATAM in the evaluation of government-sponsored reference architecture for a ground based command and control system (Gallagher, 2000). The author mentions that ATAM increases the system developer’s probability that a system built conforming to the architecture will meet the needs of its customer base. In addition, benefits of performing ATAM are summarized in these points (Gallagher, 2000): “early identification of risks, sensitivity points, and tradeoffs before design decisions are made and become costly to change”. Gallagher suggested using a program to do the ATAM-based evaluation. The evaluation pointed out that more work is needed to ensure correctness of the interfaces and integration of the components. It also revealed potential deficiencies that may have taken months, perhaps years, to uncover at a greatly increased cost to the acquirer.

Authors of the book “Software Architecture in Practice” (Bass et al., 2003) present software architecture in a real-world setting, reflecting both the opportunities and constraints that companies encounter. In addition, case studies describing architectures illustrate key points of both technical and organizational discussions.

On the other hand, application of ATAM to a multiagent system (MAS) architecture for an AGV transportation system was a valuable experience (Boucké et al., 2006). It revealed the importance of business drivers for architectural design. Especially, it improved understanding of the quality attributes and the other stakeholders improved their understanding of the fundamental architecture of the system and the important design decisions. But some critical notes have been identified by the author (Boucké et al., 2006) such as: “coming up with a utility tree proved to be difficult, time consuming, and at times tedious. A lack of experience and clear guidelines of how to build up such a tree hindered and slowed down the discussion.”

In (Venckeleer, 2006) great emphasis was put on architecture explication and the specification of architectural quality goals (architectural styles which should meet quality attributes). Functionality was largely ignored and business drivers were the starting point of the elicitation process. There was also a strong focus on implication of stakeholders during all steps of ATAM process. However, stakeholder involvement in “Phase 2” may not be realistic because it was difficult to have a common pool of questions from stakeholders for analyzing each quality attribute over architecture. Another aspect discovered during the application of ATAM is related to naming scenarios and quality attributes. In fact, results of the analysis are dependent on the selection of the scenarios and their relevance for evaluating the architecture. Future work is needed to evaluate the effects of its various usages and to create a repeatable method based on repositories of scenarios and elicitation questions.

Table 3.2 summarizes strengths/weaknesses identified during case studies application of each method. It is important to mention here that ATAM is the only method which has proven its usage in industry by its working group (Jones, 2001); (Gallagher, 2000); (Bass et al., 2003); (Boucké et al., 2006) and (Venckeleer, 2006). For the other methods (MOQARE, FDAF,

IESE NFR and Soft Goal Notation), there is no information about their usage in industry. Only case studies have been provided.

Table 3.2 Strengths and weaknesses of QRs management methods

QRs management methods	Strengths	Weaknesses
FDAF framework (Dai et al., 2005 and 2006) and (Cooper et al., 2004)	1. The RBAC aspect provides architects with the concrete information about addition of a security aspect in their application. 2. Rapide's analysis tool supports architects with detailed analysis of the system's behavior simulation at the architectural level; 3. The FDAF resource utilization aspect analysis provides architects with detailed analysis information about which component is the bottleneck (overloaded and busy all the time) and refine the UML architecture to meet the NFRs.	1. Limitations of the Alloy's analysis tool in this area: it doesn't provide modeling constructs to support the description of component's behaviour and connections; 2. Limitations of the Rapide's analysis tool and difficulty to obtain useful information from the raw data (response time analysis results presented in the graphical browser) as the number of simulated events increases; 3. Limitations of the Armani's analysis tool in this area: the mathematical assumptions restrict the systems they are modeled; 4. The Armani tool does not calculate automatically the property "sOverloaded" instead it allows changes to it.
IESE NFR/ASPIRE (Doerr et al., 2005)	1. Identifies early conflicting requirements with the use of the analysis dependency; 2. Enhances communication between stakeholders (requirements engineer, developer and customer); 3. Elicits important missed NFRs.	1. The dependency graph is used to represent dependencies between quality attributes. Graph is not used to capture NFRs (they are placed in the requirements documents template); 2. The requirement management support is performed in iterations which will be costly at long term; 3. Major rework in architecture to integrate NFRs; 4. Much rework is required during integration of the functional and NFRs through iterations.

Table 3.2 Strengths and weaknesses of QRs management methods (follow)

QRs management methods	Strengths	Weaknesses
Soft Goal Notation: Credit card system (Chung et al., 1994)	1. The framework studies covered a variety of NFRs, a number of application areas and systems with a variety of characteristics; 2. Allows to represent the relevant concepts and methods for dealing with NFRs during the software development process; 3. Links the design decisions back to the source NFRs.	1. There is a need for definition and use of more specialised methods requiring additional expertise; 2. Further work is needed towards a more rigorous evaluation of the Framework. This would involve real studies across a spectrum of developers and on a variety of different types of systems; 3. There is a need for larger bodies of goals, methods and tradeoffs to see if they can be accommodated and graphically represented; 4. There is not a closely real work with development teams from the organisations.
MOQARE: Uveitis Database (Hermann et al., 2007a and 2007b)	1. The tree structure helps to structure the elicitation process and interviews; 2. The checklists were helpful in avoiding concentration on only a few QAs, types of threats or misusers; 3. The method guides stakeholders by a process and support the reuse of knowledge by checklists and templates.	1. A domain-specific wording is preferred instead of general items in the checklist (for example user should be replaced by a specific role "nurse"); 2. The process became difficult to apply when iterations augment and hence the misuse tree became more complex : not all quality goals could be analyzed; 3. How about countermeasures which are not selected, are they analyzed further? Or omitted? 4. As How to integrate the results of MOQARE into the FRs specification document;
ATAM:Purchase2 Pay.com and MAS architecture for an AGV transportation system (Jones, 2001); (Gallagher, 2000); (Bass et al., 2003); (Boucké et al., 2006) and (Venckeleer, 2006)	1. Forces an articulation of specific quality goals; 2. Strong focus on & direct involvement of stakeholders; 3. Forces concrete consideration of business drivers; 4. Improves importance of software architecture in software engineering.	1. Quality attribute workshop is difficult and time consuming; 2. No connection to the business goals; 3. Applying ATAM requires more planning/understanding; 4. No common pool of questions for analysing each quality attribute over architecture; 5. There is a need to investigate how domain knowledge and degree of expertise affect the coverage of selected scenarios.

Table 3.3 describes the assessment of QRs management methods according to characteristics and criteria established in chapter 1 (section 1.4.13) which are: identification, decomposition, definition, representation, conflict analysis, documentation, quality standard used, and integration with FRs. QRs management methods are evaluated by their extent to address each criterion. As illustrated by the table, one can argue that most of the concepts (identification, decomposition, conflict resolution, documentation, derivation from business goals and integration with functional requirements) are not applied by these methods (Tables 3.3 and 3.4). The "Representation" and "Definition" concepts are easily addressed in these methods but "Documentation" and "Consensus on quality definitions" are absent. The "identification", "Conflict resolution", "Derivation from business goals" and "Integration with FRs" concepts are neither applied nor mentioned in the case studies neither are they described. For the used ISO/IEC quality standard, only IESE NFR method indicates in the case study the use of ISO/IEC 9126. Table 3.4 establishes comparisons of method artifacts used during these case studies. In fact, used artifacts are defined in the case studies but they need to be further described to be understandable. There is also a need for more easily applied techniques to be acceptable to users.

Table 3.3 Assessment of QRs management method's applicability

QRs management methods	Characteristics and criteria									
	Identification of quality attributes	Decomposition	Definition	Conflict analysis	Representation	Documentation	Consensus on quality definitions	Quality standard	Derivation from business goals	Integration with FRs
MOQARE	Partially	Partially	Yes	No	Yes	No	No	No	Partially	No
IESE NFR	Partially	Partially	Yes	Partially	Yes	No	No	Yes	No	Partially
ATAM	No	No	Yes	Yes	Yes	No	No	No	Partially	No
SOFT GOAL NOTATION	No	Partially	Yes	Partially	yes	No	No	No	No	No
FDAF	No	No	Yes	No	yes	No	No	No	No	No

Yes: The concept is well applied.

No: concept is not applied.

Partially: concept is mentioned in the case study but not described.

Table 3.4 Comparisons of applied QRs management methods through their artifacts

QRs management methods	Characteristics and criteria									
	Identification of QAs	Decomposition	Definition	Conflict analysis	Representation	Documentation	Consensus on quality definitions	Quality standard	Derivation from business goals	Integration with FRs
MOQARE	Interviews	Questions	Quality goals		Misuse tree				Checklist	
IESE NFR	Prioritized questionnaire	Quality model	QAs	Dependency analysis	Quality model			ISO/IEC 9126		Consolidation step
ATAM			QAs		Utility tree				Questions	
SOFT GOAL NOTATION		Soft goal graph	NFRs		Goal graph structure					
FDAF			Aspects		Extended UML					

3.1.2 Conclusion

This section presented and discussed the applicability of five QRs management methods (in case studies) classified according to three major concepts (business goals, aspect and goals oriented). Analysis and discussion of their applicability (by case study) have been described and their strengths and weaknesses have been identified.

The following sections describe QRs situation in industry and analyze the resulting indicators from industrial and academic environments.

3.2 Quality requirements management in an industrial environment

This section presents an overview of the current situation of QRs engineering practices in an industrial environment. A questionnaire was developed and distributed in industry to obtain indicators about the QRs practices in industry (Annex I). The first part of this section describes data collected from the questionnaire, the second part analyses the collected data and provides the results illustrating the real situation of QRs engineering practices in companies and their critical needs.

3.2.1 Data collection of quality requirements

The questionnaire is structured as follows: section 1 presents the purpose of the questionnaire and questions on the personal profile of each of the domain representative who complete the questionnaire. The next section describes instructions related to how answers should be formulated. Finally, the main items related to stakeholders, processes, methods, standards and the company are described. A pivot table tool (Excel 2003) is used for sorting and summarizing the collected data. A detailed description of the questionnaire is presented in Annex I.

For each section of the questionnaire, items are filled out according to the following closed-type questions: “Yes”, “No”, “Partially” or “Do not know”. The objective is to have indications about management of software QRs in the industrial environment. Some questions may require additional justification.

3.2.2 Performing the data collection process

The questionnaire was filled out by eight domain representatives from industry along with their comments. These domain representatives are practitioners in industry with different profiles and more than 3 years experience in the software quality field (Table 3.5). Two of the domain representatives have solid backgrounds in software quality engineering (11 and 20 years). Their major responsibilities are focused in process engineering and software planning.

Table 3.5 Responsibility and duration of working of domain representatives

Sum of Duration of working	Responsability1							
Position	Design Specification Programming and Test	Design of software	Planning design and test of software	Planning of software	Process Engineering	Programming and Test	Test of software	Grand Total
Architect	3							3
Developer						1		1
Evaluator							3	3
Quality assurance manager				11			5	16
Quality engineer		3			20			23
Research And Development			5					5
Grand Total	3	3	5	11	20	1	8	51

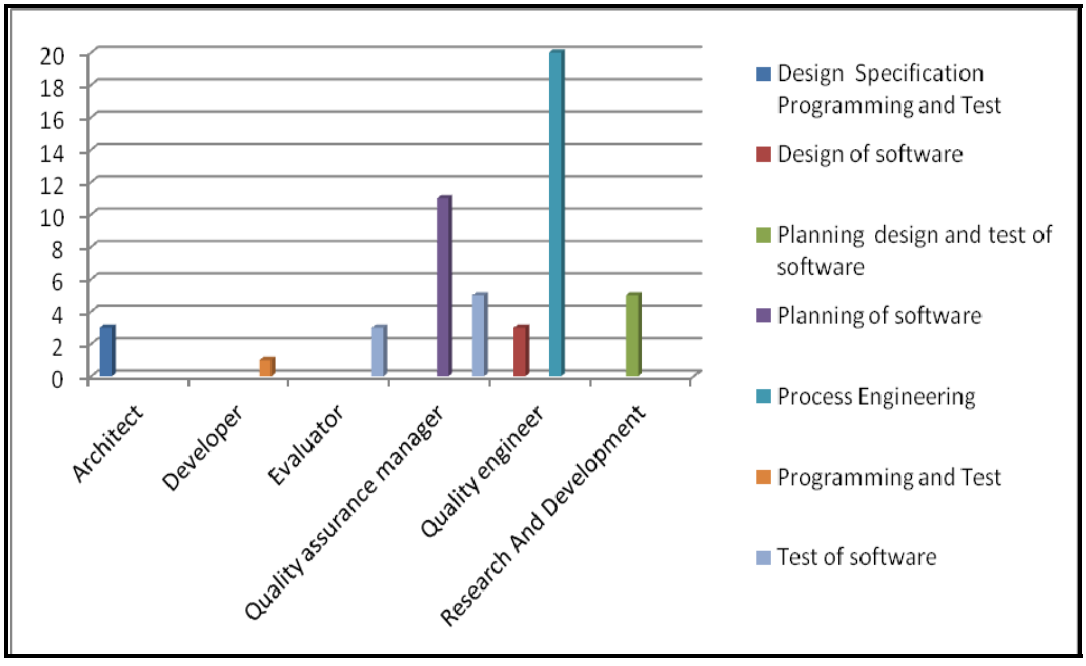


Figure 3.2 Profile of domain representatives

3.2.3 Analyzing the collected data

The present section deals with the analysis of the results collected from the survey. The following sections have been analyzed:

1. Companies and stakeholders interested by the processing of QRs of the software product;
2. Processes with QRs of the software product;
3. Methods of QRs processing of the software product;
4. Software quality engineering standards of the software product used in industry.

a) Companies and stakeholders

In this section, companies interested by the processing of quality requirements are of medium size (51-3000 people) (Table 3.6).

Table 3.6 Size of companies

Count of Size of company	Size of company				
Position	>5000 people	10 - 50 people	301 - 1000 people	51 - 300 people	Grand Total
Architect			1		1
Evaluator		1			1
Quality assurance manager				2	2
Quality engineer	1				1
Research And Development				5	5
Grand Total	1	1	1	7	10

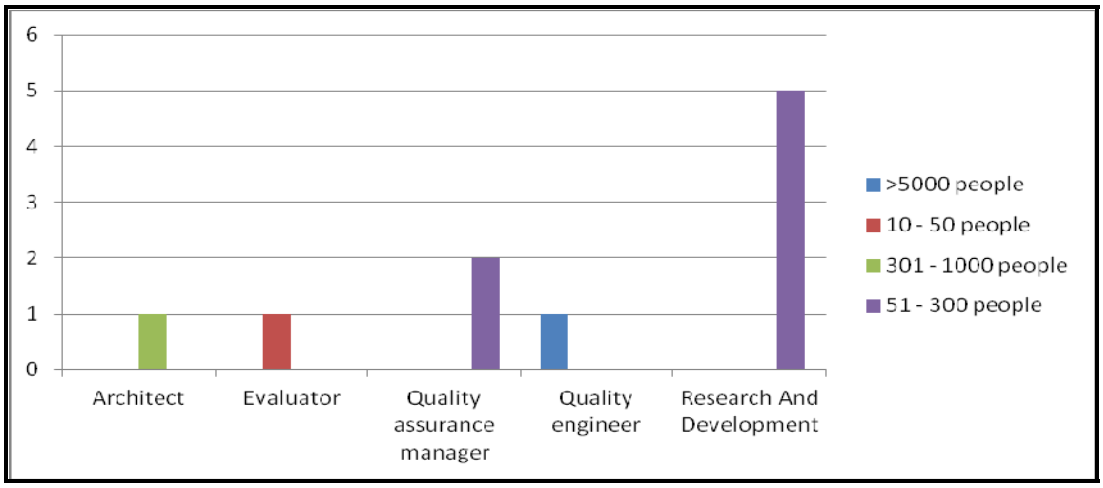


Figure 3.3 Size of companies interested in QRs processing

Their largest activity domains are: banking, electronics and logistics. But the respondents have mentioned that education, information and communication technology, government, health and banking are also important activity fields (Table 3.7).

Table 3.7 Activity domains

Count of Importance of activity domains	Importance of activity domains			
Activity domains	Largest part	Most important	Not relevant	Grand Total
Aeronautics			1	1
Banking	1	1		2
Education		2		2
Electronics	1			1
Government		1		1
Health		2		2
Information & communication technology		1		1
Logistics	1			1
RS&D		1		1
Grand Total	3	8	1	12

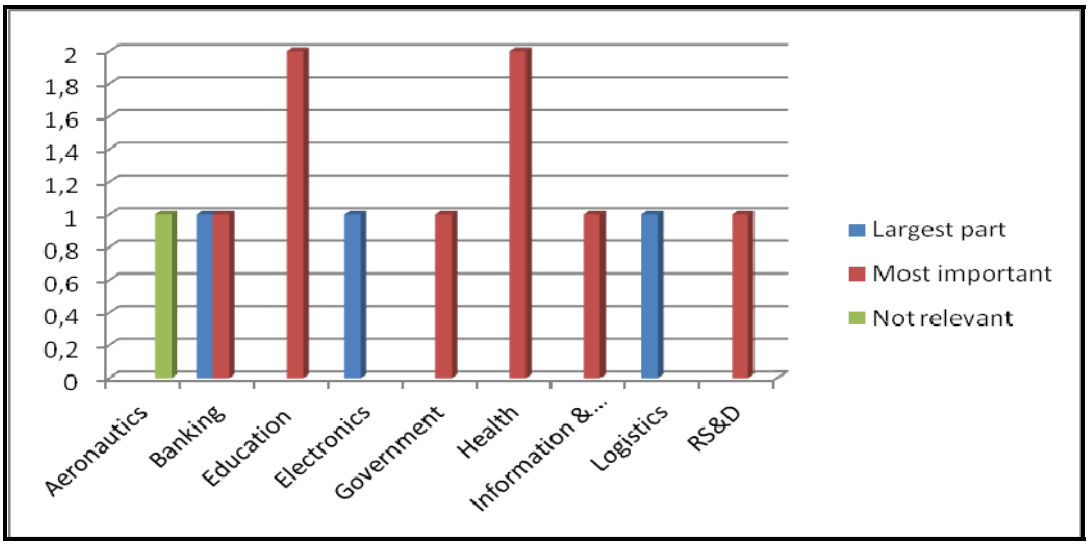


Figure 3.4 Activity domains of companies

The most important type of projects and software developed by the company are in systems, business and internet (Table 3.8). The moderately important types of developed projects are the embedded and systems ones.

Table 3.8 Developed projects

Count of Importance of projects	Importance of projects			
Projects developed	Moderatly important	Most important	Not relevant	Grand Total
Business		2		2
Embedded	1			1
Internet based		2		2
Real time			1	1
Scientific			1	1
Systems	1	1		2
Test & Test training			1	1
Grand Total	2	5	3	10

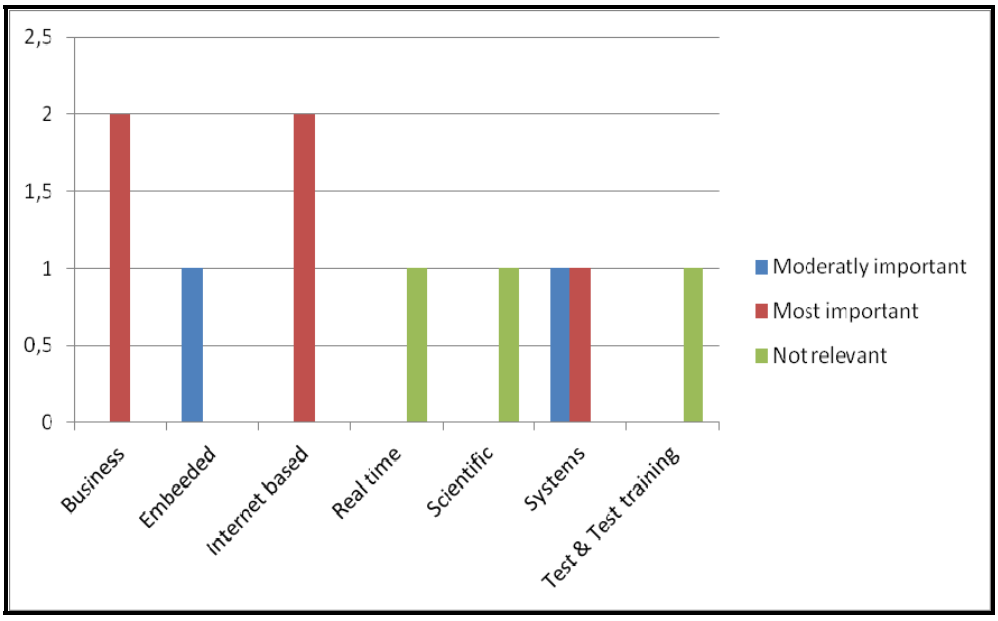


Figure 3.5 Importance of developed projects

Developed projects are almost not critical for the company as shown in Table 3.9 and Figure 3.6, except for those developed in business, internet and test training.

Table 3.9 Critical level of developed projects

Count of Business critical level	Business critical level		
Projects developed	Critical	Not critical	Grand Total
Business	1	1	2
Embedded		1	1
Internet based	1	1	2
Personal		1	1
Real time		1	1
Scientific		1	1
Systems		2	2
Test & Test training	1		1
Grand Total	3	8	11

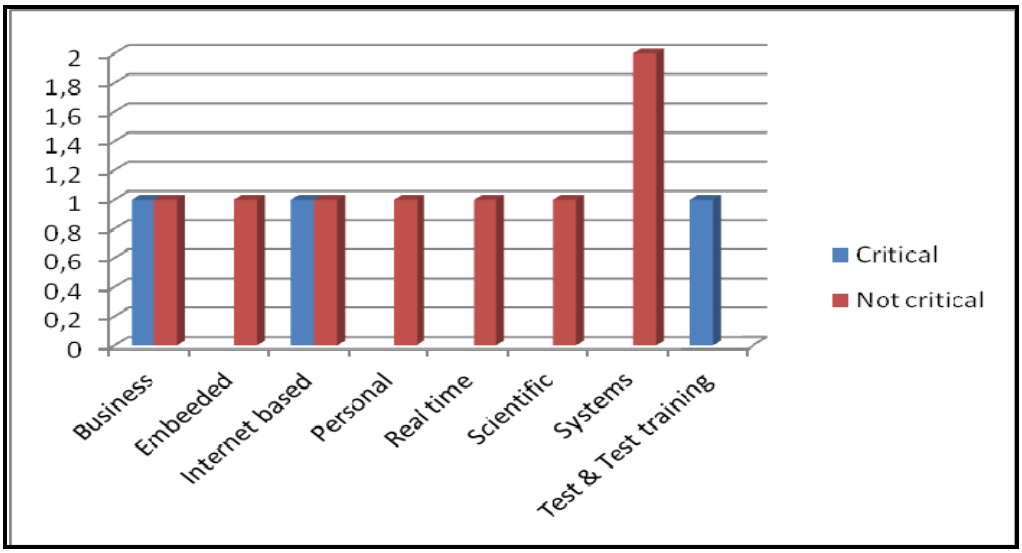


Figure 3.6 Critical level of developed project

Most of the stakeholders interested by the QRs processing are: IT department or business operations and department of management. Persons responsible for managing QRs for a specific software development project are: project managers, quality engineer and quality assurance manager. Most would have at least 2 years experience. Project and test managers have more than 5 years experience in their respective fields (Table 3.10).

Table 3.10 Interested stakeholders by QRs

Count of His experience	Who is responsible for QRs for a SWD project		His experience										
	Development manager	Development manager Total	Project manager		Project manager Total	quality assurance manager	quality assurance manager Total	Quality engineer		Quality engineer Total	Software & system developer	Software & system developer Total	Grand Total
Department interested for QRs	1 year & more		1 year & more	5 years		1 year & more		1 year & more	5 years		1 year & more		
All			2		2								2
Department of IT or business operations	1	1						7		7			8
Department of management			1	1	2	1	1		1	1	1	1	5
Grand Total	1	1	3	1	4	1	1	7	1	8	1	1	15

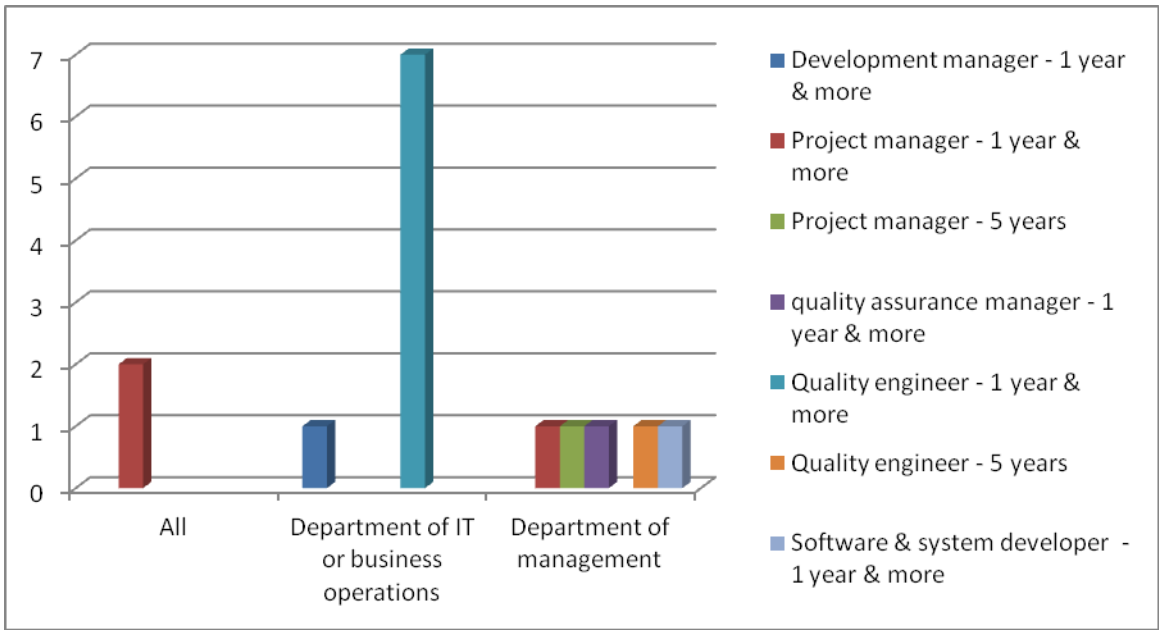


Figure 3.7 Stakeholders and their experience

Training in processes and methods is a priority for project managers and software and system developers. Norms and standards are also important for quality engineers and finally, more software tools should be available to software and system developers (Table 3.11).

Table 3.11 Type of training

Count of Type of training given in software quality	People involved				
Type of training given in software quality	Project administrator	Project manager	Quality engineer	Software & system developer	Grand Total
Norms & standards	1	1	2		4
Processes & methods		4	1	2	7
Software tools				2	2
Grand Total	1	5	3	4	13

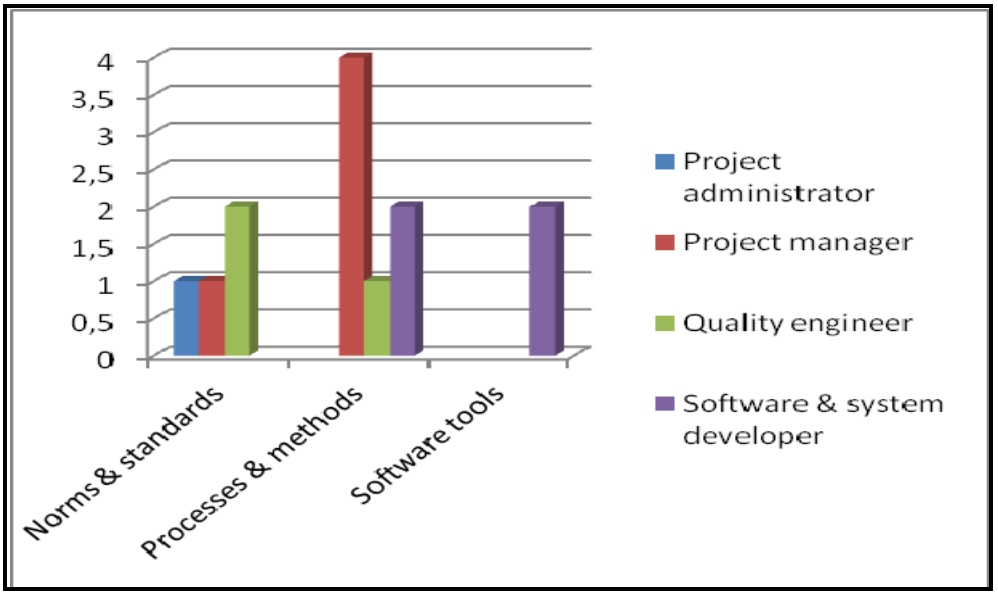


Figure 3.8 Stakeholders and their experience

b) Processes

Figure 3.9 indicates that most of the organizations use a QRs process where identification and specification activities of QRs are the most important. Prioritization and documentation are the next most important activities. Finally, representation of QRs is reported in the third position. One notes that traceability of QRs is also an important activity to be taken into account in the quality process.

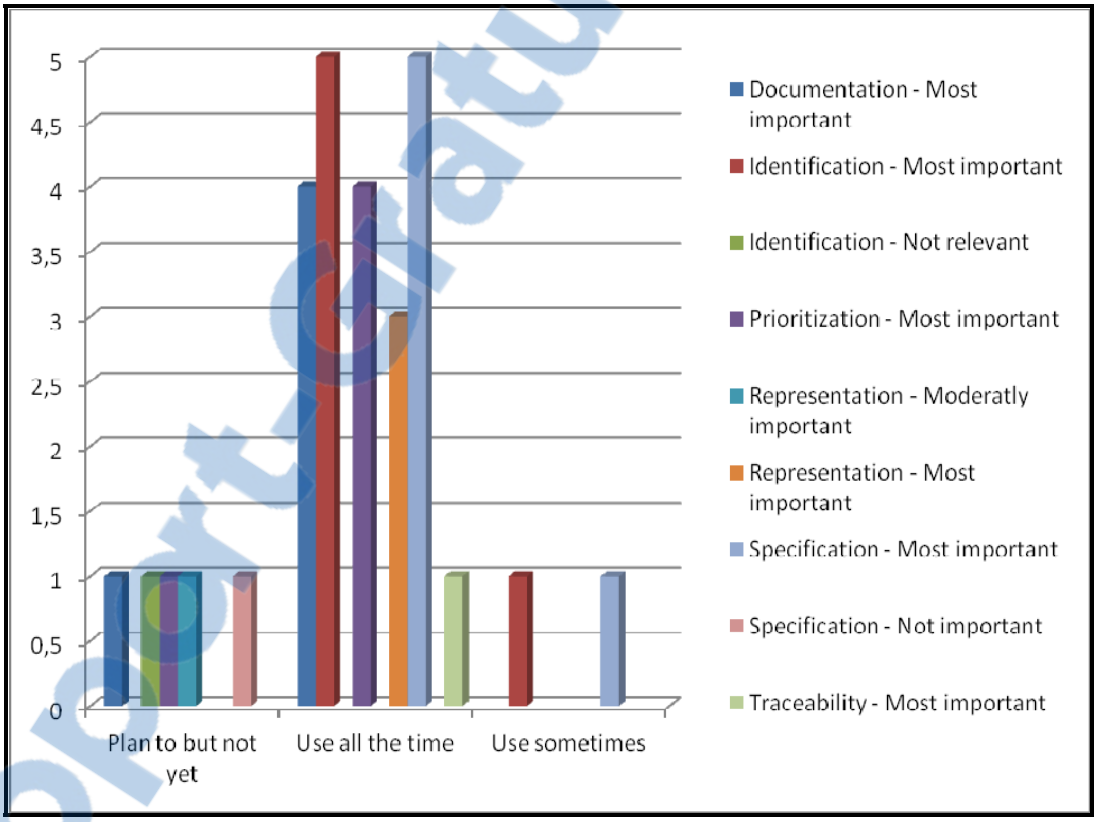


Figure 3.9 QRs process activities

In Table 3.12, 40% of the responses indicate an absence of software tools supporting the quality requirements process. Some respondents mentioned the use of "HP Quality Centre", "Rationale" and other market standard software.

Table 3.12 Type of Sofware tools

Count of QRs tools	
QRs tools	Total
Other market standard software	10,00%
HP Quality Centre	10,00%
N/A	20,00%
Rationale Software Inc.	20,00%
We do not use a software tool	40,00%
Grand Total	100,00%

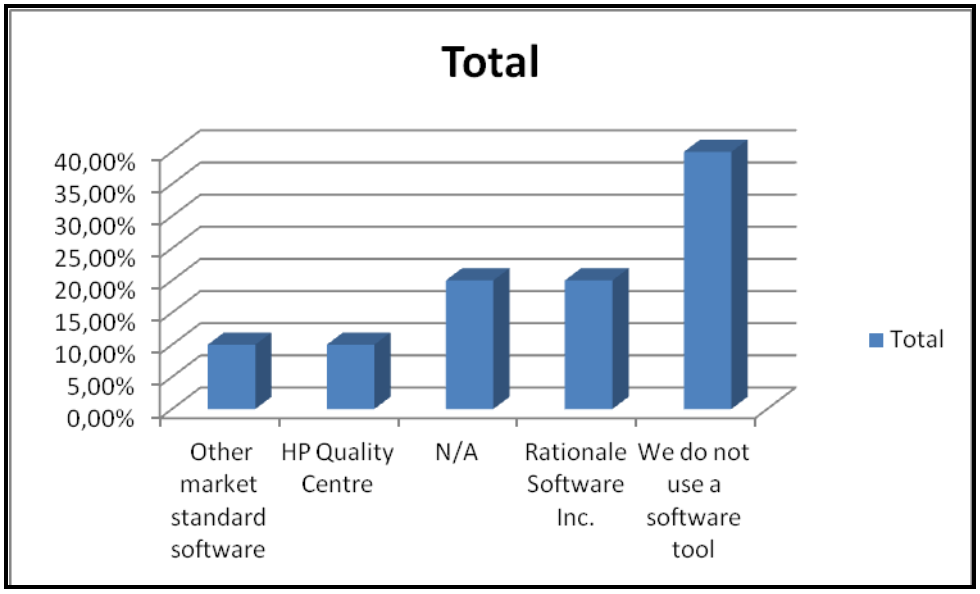


Figure 3.10 The use of software tools

Finally the critical need for a structured and well defined quality requirements process is strongly desired (88%), as seen in Table 3.13.

Table 3.13 The need to improve quality

Count of Improvequality	
Improvequality	Total
N/A	11,11%
Yes	88,89%
Grand Total	100,00%

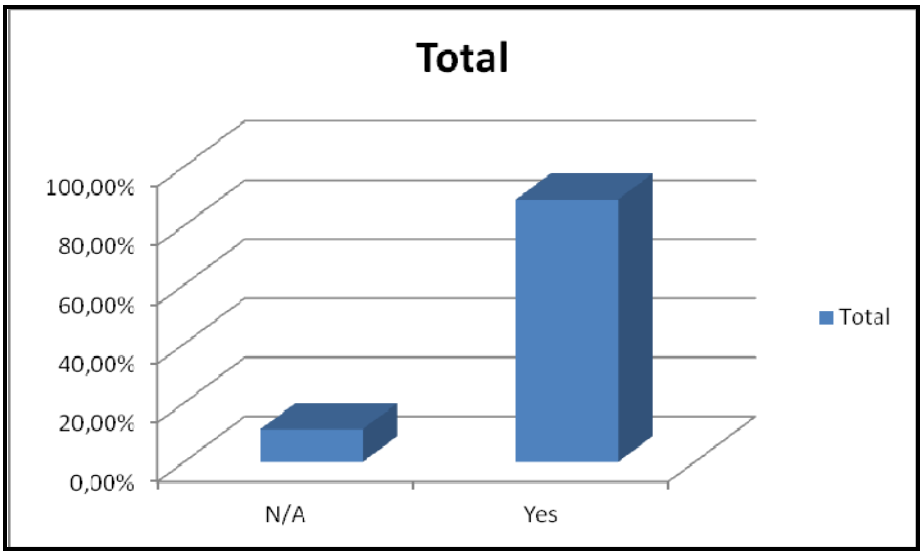


Figure 3.11 The need to a structured QRs process

c) Methods

According to the responses, "Interviews", "Meetings" and internal methods of organization are the most used techniques to identify QRs. "Brainstorming", "Observations" and "Checklists" are used in second place (Figure 3.12).

Table 3.14 Techniques to identify QRs

Count of Identification	
Identification	Total
Brainstorming	11,11%
Checklists	11,11%
Internal methods	16,67%
Interviews	22,22%
Meetings	16,67%
Observations	11,11%
Questionnaire	5,56%
We do not use any identification method	5,56%
Grand Total	100,00%

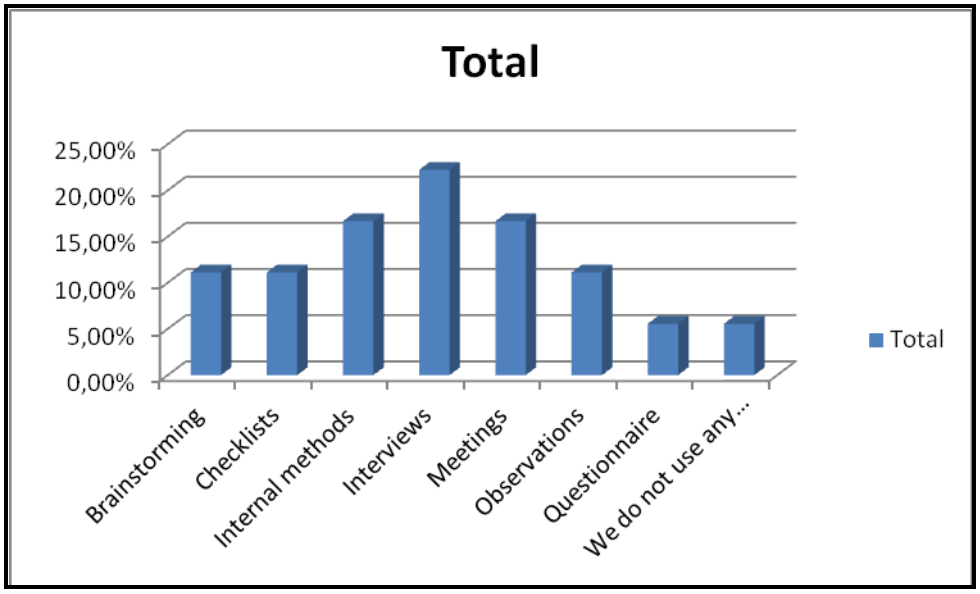


Figure 3.12 QRs identification most used techniques

For the decomposition method of QRs, there is an absence of a recognized technique (57%). The only technique used is the “Quality model”, represented by 26% of the survey (Table 3.15).

Table 3.15 Techniques to decompose QRs

Count of decomposition	
decomposition	Total
Quality model	31,25%
We do not use any decomposition method	68,75%
Grand Total	100,00%

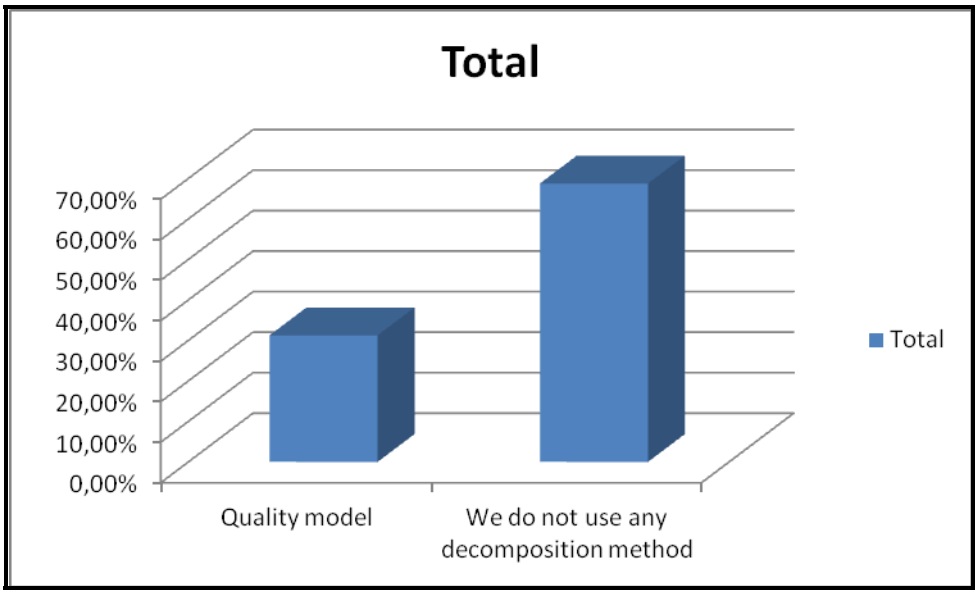


Figure 3.13 QRs decomposition most used techniques

QRs are first documented in "Template" (42%), 37% have mentioned the use of the requirements specification document (RSD), see Table 3.16.

Table 3.16 Techniques to document QRs

Count of Documentation	
Documentation	Total
RSD	43,75%
Template	50,00%
We do not use any documentation formalism	6,25%
Grand Total	100,00%

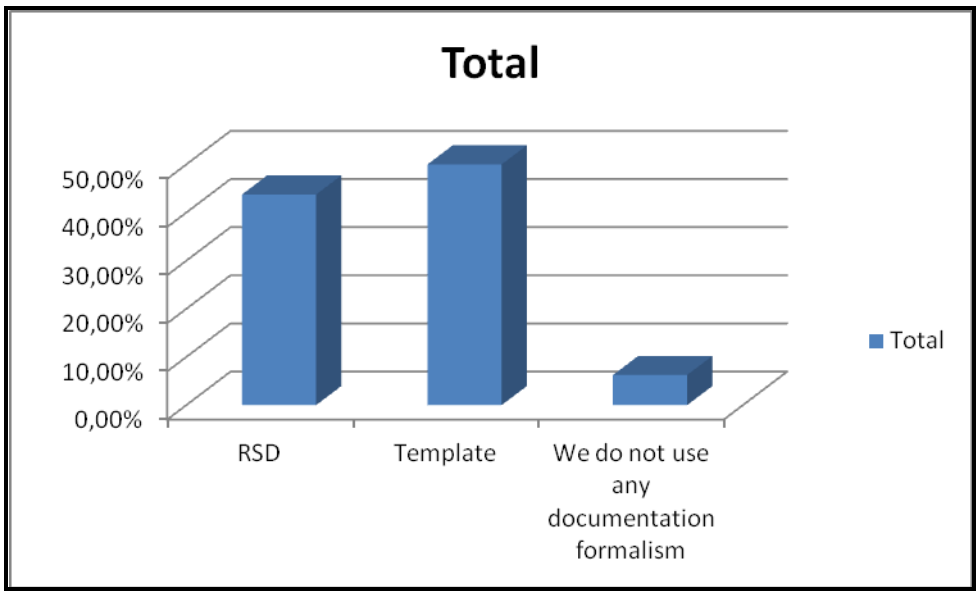


Figure 3.14 QRs documentation most used techniques

In organizations dealing with quality requirements, the size of software projects may vary from mega to big to medium (Table 3.17 and Figure 3.15).

Table 3.17 Size of software projects

Count of Size of SWP	Size of SWP				
Response no	Big 300-I000 KLOC	Medium 50-300 KLOC	Mega >1 MLOC	Small <50 KLOC	Grand Total
SW project1	1		2	1	4
SW project2	2		1		3
SW project3		2			2
Grand Total	3	2	3	1	9

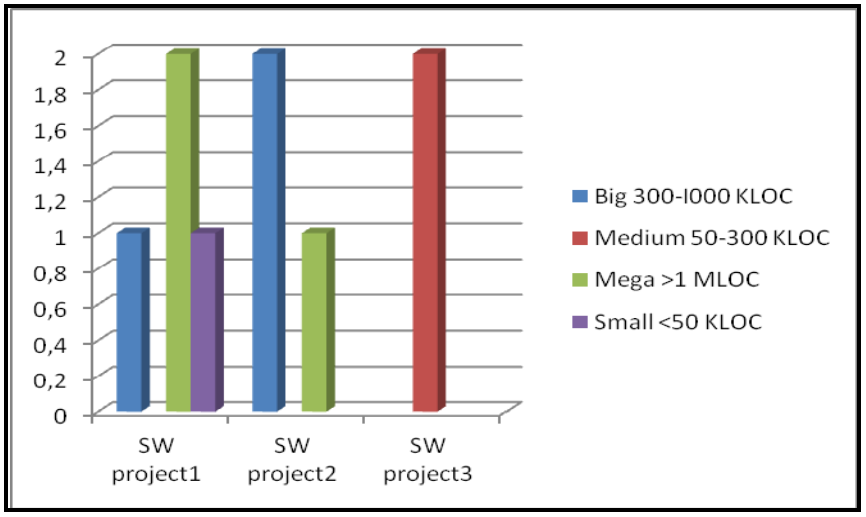


Figure 3.15 Size of developed software projects

The total effort for each type of software project is thousands-hundreds for the mega project; 40-few hundred for the big project and 8-40 participants for the medium project (Figure 3.16).

Table 3.18 Total effort of software projects

Count of Total effort of SWP		Total effort of SWP				
Response no	Size of SWP	2 persons for 1- 2 weeks	40- few hundreds	8-40 participants	Hundreds-thousands	Grand Total
SW project1	Mega >1 MLOC				1	1
	Small <50 KLOC	1				1
SW project1 Total		1			1	2
SW project2	Big 300-I000 KLOC		1			1
SW project2 Total			1			1
SW project3	Medium 50-300 KLOC			1		1
SW project3 Total				1		1
Grand Total		1	1	1	1	4

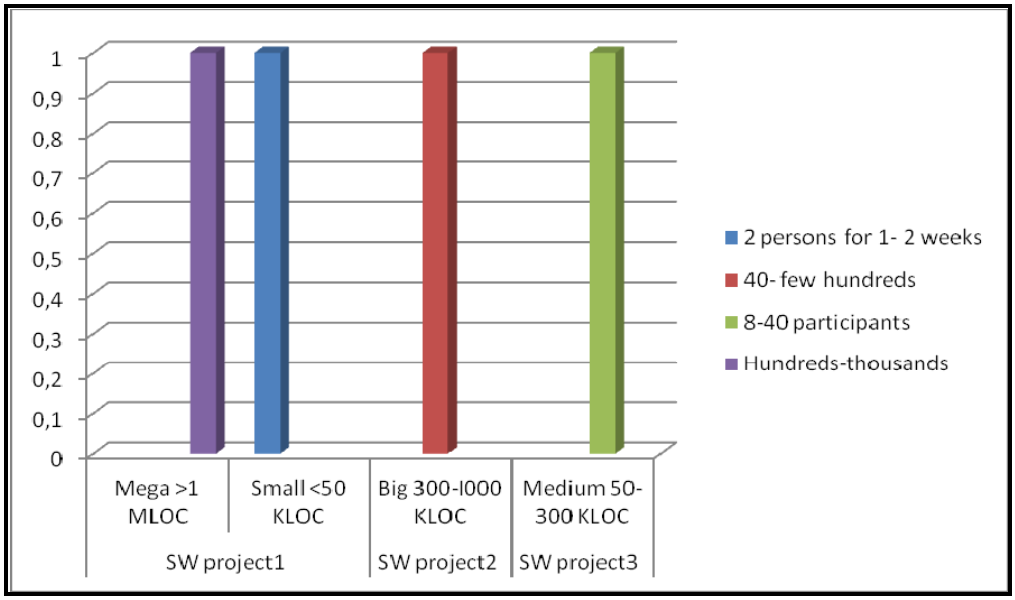


Figure 3.16 Total efforts for the developed software projects

Also, hierarchy levels for each software project vary from 1 to 4 levels for the mega project to 3 levels for the small and big projects and 2 levels for the medium project (Table 3.19).

Table 3.19 Hierarchy levels of software projects

Count of Hierarchy of authority	Hierarchy of authority				
Size of SWP	1 level	2 levels	3 levels	4 levels	Grand Total
Big 300-I000 KLOC			1		1
Medium 50-300 KLOC		1			1
Mega >1 MLOC	2			1	3
Small <50 KLOC			1		1
Grand Total	2	1	2	1	6

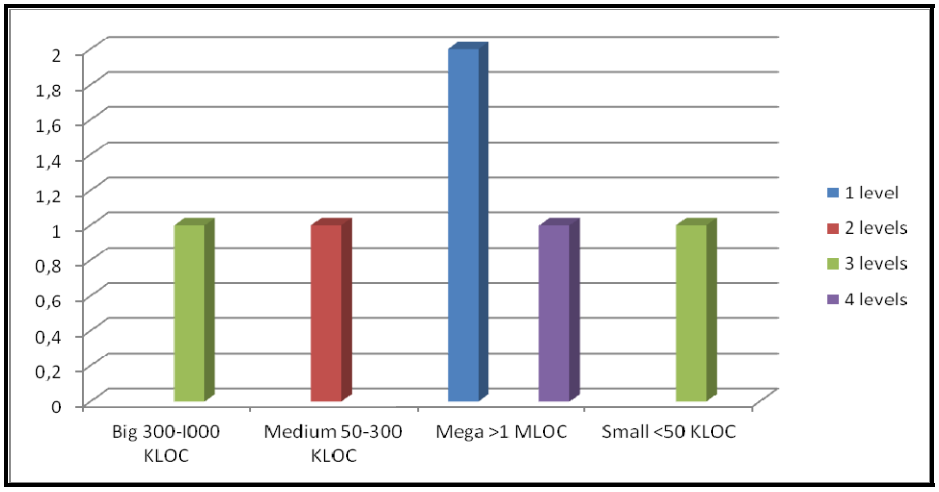


Figure 3.17 Hierarchy levels for the developed software projects

Duration of the software projects vary from (Figure 3.18):

- 2 years, 2-3 years and >5 years for a mega project
- 2 years and 3-5 years for a big project
- 2 years and 2-3 years for a medium project
- 2 years for a small project.

Table 3.20 Duration of software projects

Count of Duration of SWP2	Duration of SWP				
Size of SWP	<2 years	>5 years	2-3 years	3-5 years	Grand Total
Big 300-I000 KLOC	2			1	3
Medium 50-300 KLOC	2		1		3
Mega >1 MLOC	2	1	1		4
Small <50 KLOC	1				1
Grand Total	7	1	2	1	11

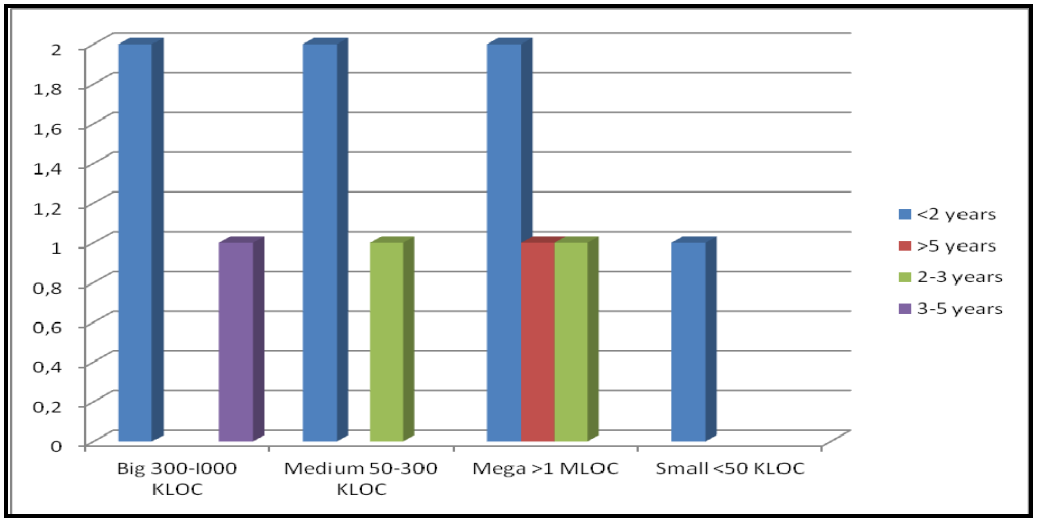


Figure 3.18 Duration of the developed software projects

d) Standards

The survey indicates that most software quality engineering standards supporting organizations are ISO/IEC 9126 and 14598. Two other standards which were suggested by respondents are ISO/IEC 25051 and 15408 (Figure 3.19).

Table 3.21 Quality standards

Count of Quality standard	
Quality standard	Total
ISO/IEC 15408	1
ISO/IEC 25051	1
IEEE 830	2
ISO / IEC 14598	3
ISO / IEC 9126	3
Grand Total	10

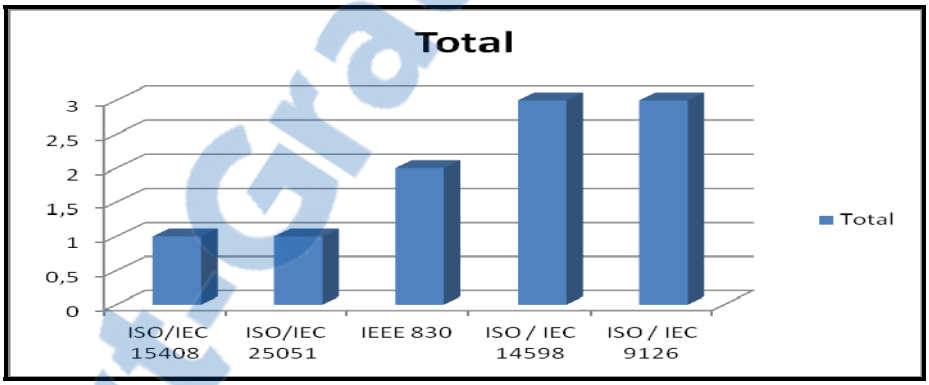


Figure 3.19 Used standards

The project and quality assurance managers are responsible for applying these standards in their organizations. They have 5 or more years experience in this field (Figure 3.20).

Table 3.22 Responsible of standards

Count of How long	How long	
Responsible	> 5 years	Grand Total
Project manager	1	1
Quality assurance manager	7	7
Grand Total	8	8

Figure 3.20 Experience related to the responsibility for standards

For organizations using ISO/IEC 9126, the used parts of this standard are quality model, internal quality, external quality and quality in use (Figure 3.21). They are used 21 times for 50 projects (Figure 3.22).

Table 3.23 Used parts of ISO/IEC 9126

Count of Parts of standard	
Parts of standard	Total
External quality	2
Internal quality	2
Quality in use	1
Quality model	3
Grand Total	8

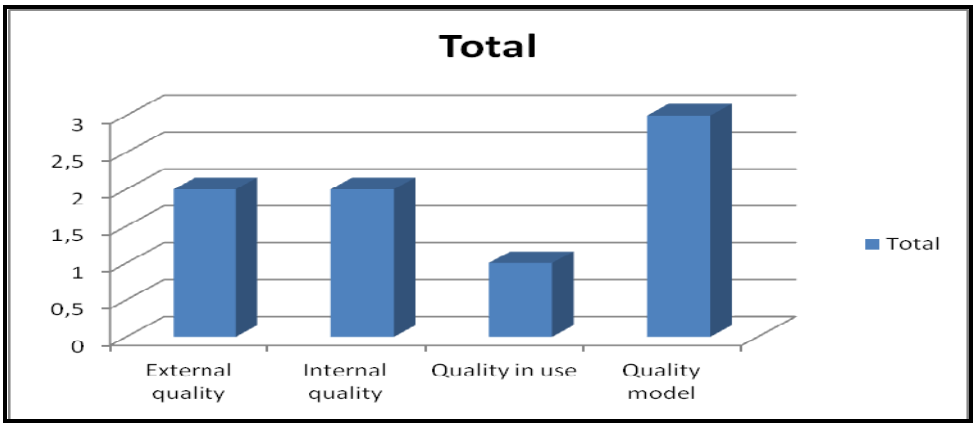


Figure 3.21 Parts of ISO/IEC 9126

Table 3.24 Frequency use of times of ISO/IEC 9126

Count of Frequency of use projects		Frequency of use projects	
Frequency of use times	Parts of standard	50	Grand Total
21	External quality	1	1
	Internal quality	1	1
	Quality in use	1	1
	Quality model	1	1
21 Total		4	4
Grand Total		4	4

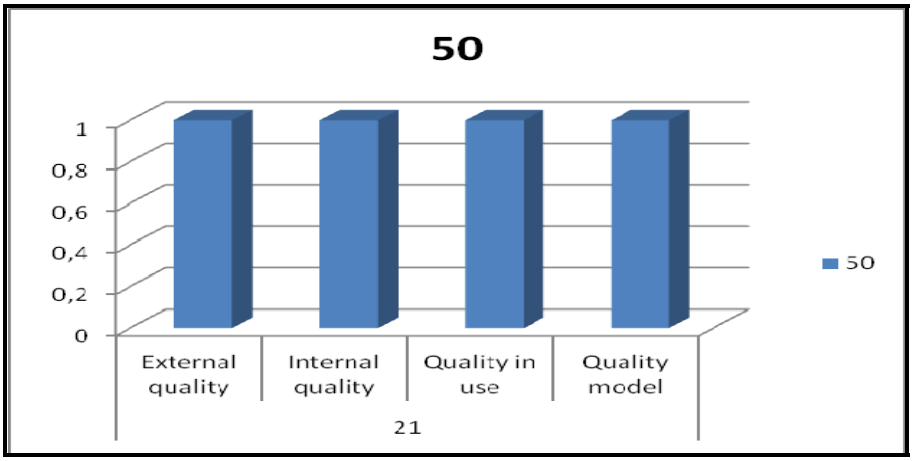


Figure 3.22 Frequency use of times of ISO/IEC 9126 per projects

e) Conclusion

Analysis of the questionnaire provides general observations about the software QRs subject. Resulted indicators are related to the motivation of organizations to have the best engineering practices of QRs, the difficulty to apply some QRs management techniques and the critical need to a structured QRs process with its supporting software tool.

3.2.3 Analysis of resulted indicators from industry and academic environments

This section analyzes obtained data from industrial and academic environments. In other words, indicators resulted from applied software QRs management methods in their case studies (section 3.1) and from collected data from questionnaires (section 3.2) are analyzed. Critical needs seen by domain representatives in industry in the field of quality requirements are identified and conclusions and justifications for the proposed solution are formulated. Resulted data will determine future requirements for the research solution design. The analysis process is carried out in the following categories:

- Identification of software QRs;
- Representation of software QRs;
- Documentation of software QRs;
- Integration of software QRs with the FRs model;
- Quality standard used.

3.2.3.1 Resulted indicators from applied QRs management methods in their case studies:

- The applied methods need to improve their process with the software QRs management techniques (identification, decomposition, conflict resolution, documentation, derivation from business goals and integration with functional requirements);
- The need for more understanding and applying quality standards;
- There is a need for understandable and applied techniques to be acceptable by users;
- Critical need for well described and understandable artifacts;
- A lack of understanding of quality attributes in the software engineering community (the same interpretation of the quality attribute with different attribute names);
- Difficulty to define a unique terminology of QAs among stakeholders;
- Importance of interaction and consultation with domain people to capture priorities for requirements and to resolve terminology problems;
- A lack of contact with domain people during the case study;
- A need for a clarifying technique of the meaning of QAs;

- A critical need for documenting QRs and integrating them in the RSD.

3.2.3.2 Analysis of collected data about QRs engineering practices in industry has provided the following indicators:

- QRs represent an interesting domain field and an important aspect to be addressed in organizations;
- Most of the organizations use a software QRs process where identification, specification, prioritization, documentation and representation activities of QRs are the most important (Figure 3.9);
- The need for more software QRs engineering practices (decomposition techniques are either partial or absent (Figure 3.13) as is traceability (Figure 3.9));
- "Interviews", "Meeting" and internal methods are the most used techniques to identify QRs (Figure 3.12);
- The QRs process needs to be supported by software tools (Figure 3.10);
- The need for training in quality processes, norms and standards and software tools;
- Critical need for a structured and well defined quality requirements process (88%) (Figure 3.11);
- Use more software quality engineering standards (Figure 3.19):
  - ISO/IEC 25051 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing (this International Standard is applicable to COTS "Commercial Off-The- Shelf" software products.
  - ISO/IEC 15408 – Evaluation Criteria for Information Technology Security (represents the outcome of a series of efforts to develop criteria for evaluation of IT Security that are broadly used within the international community).

3.2.3.3 Conclusions and justifications for the proposed solution

In conclusion, the resulted indicators from industry show the existence of an interest for the QRs domain field where most of the organizations use a software QRs process. However, this process needs to be:

1. Improved by more structured software QRs management techniques;
2. Supported by more software quality engineering standards and tools.

Resulted indicators from applicability of QRs management methods in their case studies show that QRs engineering techniques need to be adequately applied, appropriately used and easily understandable. There is also a need for methods to detail meanings of the QAs, document and integrate them in the RSD document. Finally, there is a need to easily apply the software quality engineering standards and to use a unified terminology of QAs among stakeholders.

From the previous resulted indicators, future requirements of the proposed research solution are summarized in developing new techniques for:

- Identifying and defining software QRs ;
- Representing software QRs and describing their traceability;
- Resolving conflicts among them;
- Documenting software QRs in a specific format such as a template.
- Integrating software QRs with the FRs model;

3.3 Innovative aspects of the proposed research solution: SOQUAREM (SOftware QUALity Requirements Engineering Method)

In this section, innovative aspects of SOQUAREM method are highlighted by describing its specific features, meta-model, building process and process structure.

3.3.1 Specific features of SOQUAREM method

SOQUAREM solution is proposed to palliate some of the limitations of the software QRs management methods. It addresses the list of QRs managing criteria (Table 3.25). Its innovative aspects are represented as follows:

1. More interaction with stakeholders and domain experts during consensus and free dialogue sessions;
2. Use of intentional modeling and motivation of business in the derivation process of quality attributes;
3. Structured derivation of quality goals from business goals by using Business Context Table (BCT) and Business Motivation Model (BMM). Derivation step of quality attributes from business goals is fully described in SOQUAREM;
4. Use of scenarios at the requirements level to resolve terminology problems and infer the right quality attribute;
5. Use of transformation rules which are: statement rules to define business goals, refinement rules to refine business goals, linkage rules to derive quality attributes from business goals and mapping rules to link quality attributes to the FRs model;
6. Use of ISO/IEC SQuaRE 25030 as supporting quality standard for SOQUAREM process;
7. Use of a quality template to specify and document quality attributes;
8. Use of prioritizing methods (impact matrix and weighted method) to resolve conflicts among quality attributes.

Table 3.25 SOQUAREM characteristics

Requirements for the research solution	Characteristics and criteria							
	Identification	Derivation from business goals	Definition	Representation	Conflicts analysis	Documentation	Quality standard	Integration with FRs
SOQUAREM	BCT BMM Consensus session Statement and refinement rules	BCT BMM Linkage rules	QAs	Utility tree Scenarios template	Impact matrix Weighted method	Template	ISO/IEC SQuaRE 25030	Mapping rules Scenarios template

3.3.2 Meta-Model of SOQUAREM method

SOQUAREM represents an intentional, scenarios-oriented approach to quality requirements engineering. Modeling elements in SOQUAREM include business goals; quality attributes scenarios, actions and quality standard ISO/IEC SQuaRE 25030 (Figure 3.23). Business goals, influencer and strategies are provided from the BMM model. They could be traceable to the concepts of quality attributes, actors and actions. Quality attributes are clarified into quality scenarios where details about actions and assets related to their achievement are defined. Quality attributes are also specified by using the ISO/IEC SQuaRE 25030 quality standard as a supporting framework.

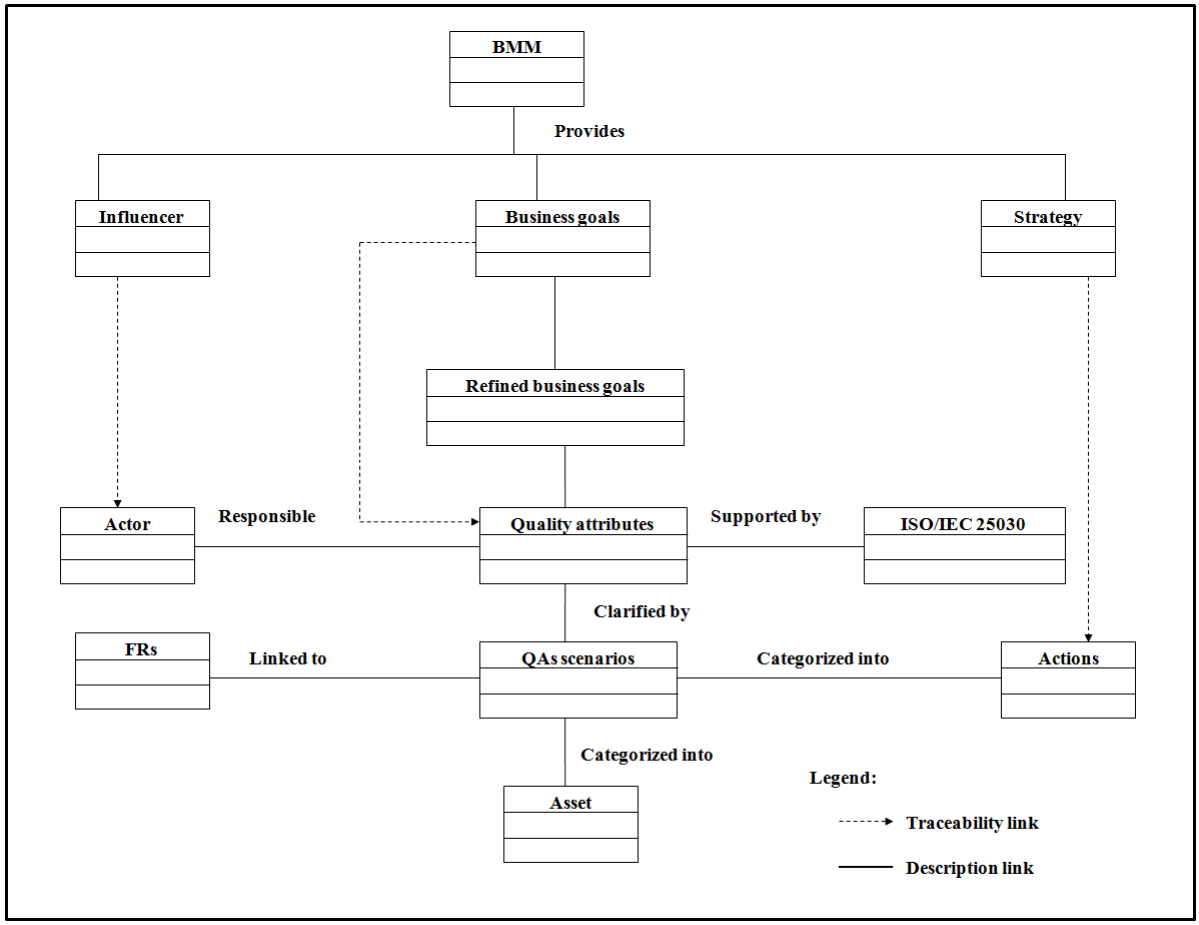


Figure 3.23 Meta-Model of SOQUAREM

3.3.3 The SOQUAREM building process

Figure 3.24 shows the SOQUAREM building process which presents mapping of concepts from different research resources (such as quality attributes template, scenarios descriptions, Business elements…), the domain experts' verification and process improvements. The dashed boxes present different authors from literature review who deal with similar concepts in their specific context. For example, scenarios descriptions in ATAM method (Kazman et al., 2000) are used to detail the meaning of quality attributes with a specific description related to an architectural context. The quality attributes template (Moreira et al., 2002) describes quality attributes with specific items to address aspectual quality attributes crosscutting with functional requirements.

Elements of context (such as business, user and software domain) help to identify and refine business goals by using BMM and BCT concepts. The quality standard ISO/IEC SQuaRE 25030 is used during the linkage process of QAs to business goals to infer the right quality attribute. Scenario descriptions are semi formal methods used to make the QAs operational and help their integration in the FRs process. Prioritization techniques (Moreira et al., 2002) are used to resolve conflicts among quality attributes. QAs template (Brito et al., 2002) and utility tree (Kazman et al., 2000) are concepts used to document and represent quality attributes.

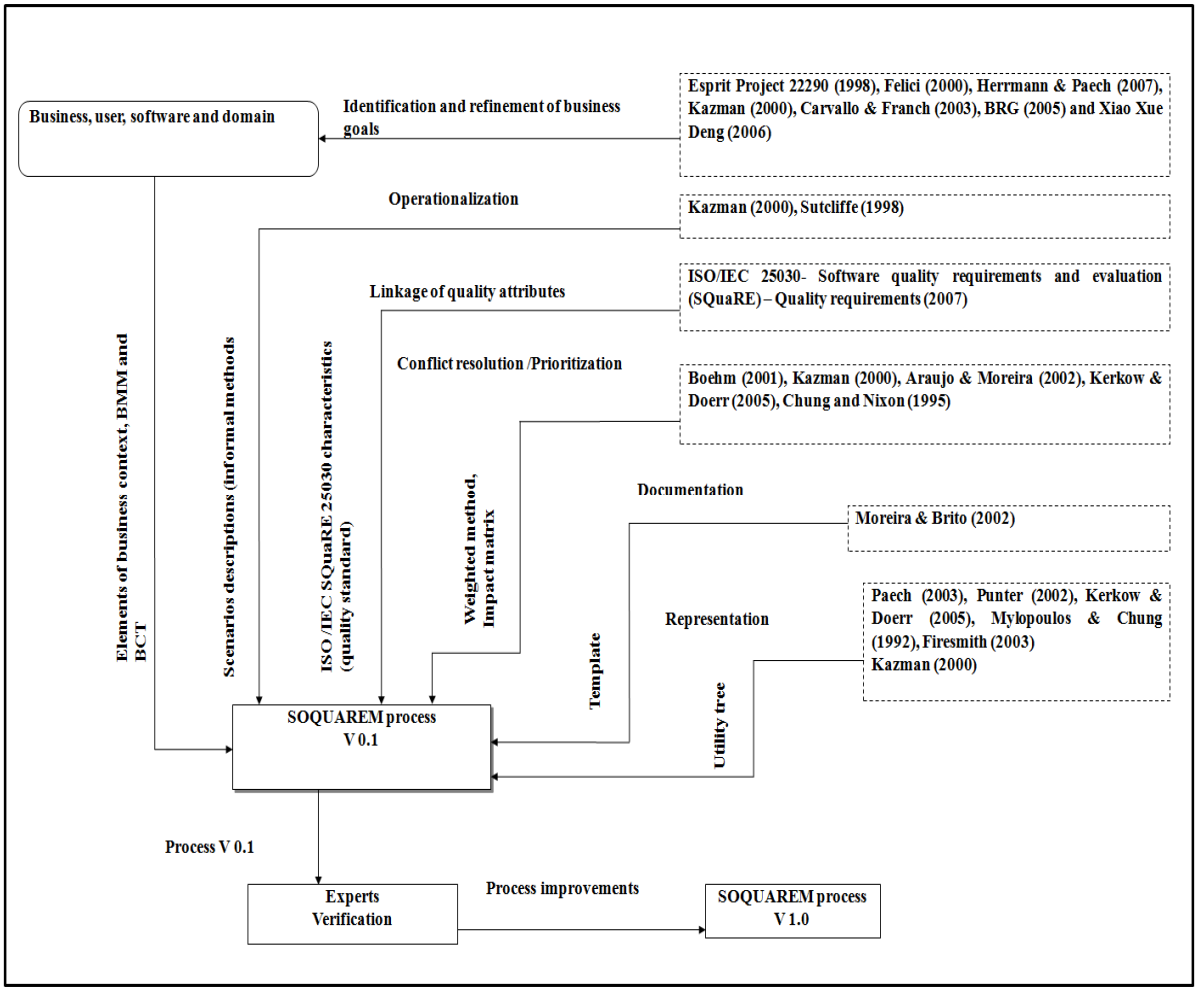


Figure 3.24 SOQUAREM building process

3.3.4 SOQUAREM process structure

The SOQUAREM structure, illustrated in Figure 3.25, is organized around phases and uses various techniques and tools (heuristics, mathematical and intentional modeling), quality standard ISO/IEC 25030 and transformation rules. Stakeholders and domain experts are involved during the process operation. Techniques used are either informal, heuristic or semi formal. The informal ones are consensus and free dialogue sessions, scenario descriptions and templates. Scenario descriptions are used to detail the meaning of quality attributes and make them operational. Heuristic techniques use descriptive methods to help clarify the business goals and identify quality attributes. Semi formal methods use UML modeling to represent the operational part of the quality attribute (actions undertaken to achieve it) and to link them to the functional requirements (represented in the use case model). Mathematical methods such as utility tree, impact matrix and weighted methods are used to represent quality attributes and resolve conflicts among them. Transformation rules are used during the whole process to regulate the operation process and are subdivided into statement rules to define business goals, refinement rules to refine business goals into refined business goals, linkage rules to derive quality attributes from business goals and mapping rules to link quality attributes to the functional process.

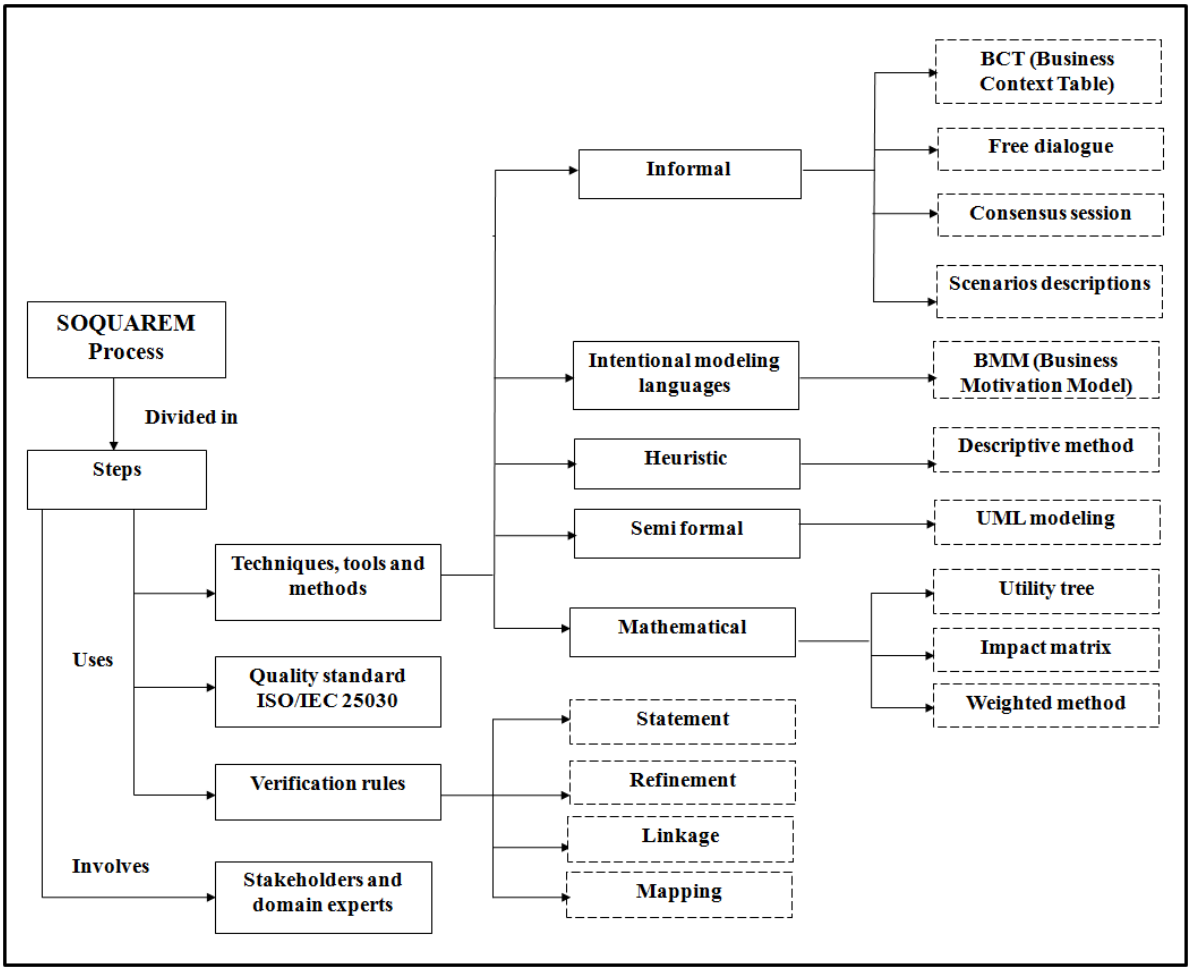


Figure 3.25 SOQUAREM process structure

3.4 Conclusion

This chapter presented the fundamentals of the research execution. Initially, applicability of existing QRs management methods has been described by analyzing their case studies and identifying their strong and weak points. Resulted analysis demonstrated that most of the applied methods do not fully apply the QRs management concepts (most of the applied concepts are mentioned in the case study but not described enough (Tables 3.3 and 3.4). QRs management methods need future work to evaluate their various usages across a large spectrum of users and systems and should be validated in concrete situations with real companies.

Subsequently, the current situation analysis of quality requirements seen by industry has been provided by developing a questionnaire. Resulted indicators pinpointed critical needs, major difficulties in addressing quality requirements and important directives for improving the QRs processing of the software product in industry.

Again, analysis of QRs situation in academic and industrial environments has been conducted. This part analyzed resulted indicators from both the questionnaire and the applied methods and provided relevant requirements for the SOQUAREM method which have been concretized in the fourth section “SOQUAREM innovative aspects”. In fact, innovative aspects of the solution have been established by describing its specific characteristics and its design and structure processes.

Chapter 4 describes in detail the proposed solution SOQUAREM: its key concepts and process model.

CHAPTER 4

SOQUAREM: SOFTWARE QUALITY REQUIREMENTS ENGINEERING METHOD

This chapter presents a detailed description of SOQUAREM (SOftware QUAlity Requirements Engineering Method) method. Section 1 introduces the high conceptual levels of SOQUAREM and its process for producing QAs list. Section 2 defines and develops its key concepts. Section 3 describes and details SOQUAREM process model. Section 4 concludes this chapter.

4.1 SOQUAREM method

The proposed method is business goals-centric; stakeholder-centered and scenario-oriented (Djouab and Suryn, 2011a). It is organized around 2 high conceptual levels (Figure 4.1):

- **The business goals level:** identifies important business goals (BGi) from the BMM model and BCT concept (next section). Specific rules are used to refine business goals. Consensus and free dialogue sessions are used to confirm the refined business goals (RBGi) with stakeholders and domain experts.
- **The system quality attributes level:** Quality attributes are derived from the business goals according to the quality standard ISO/IEC 25030 and linkage rules. They are also detailed and operationalized by using the "Scenarios template" concept. Quality attributes are analyzed for possible conflicts and consolidated by using prioritizing techniques. They are retraced to their original business goals by applying the "Utility tree" concept. Finally, quality attributes are linked to the "Use case" model by using mapping rules.

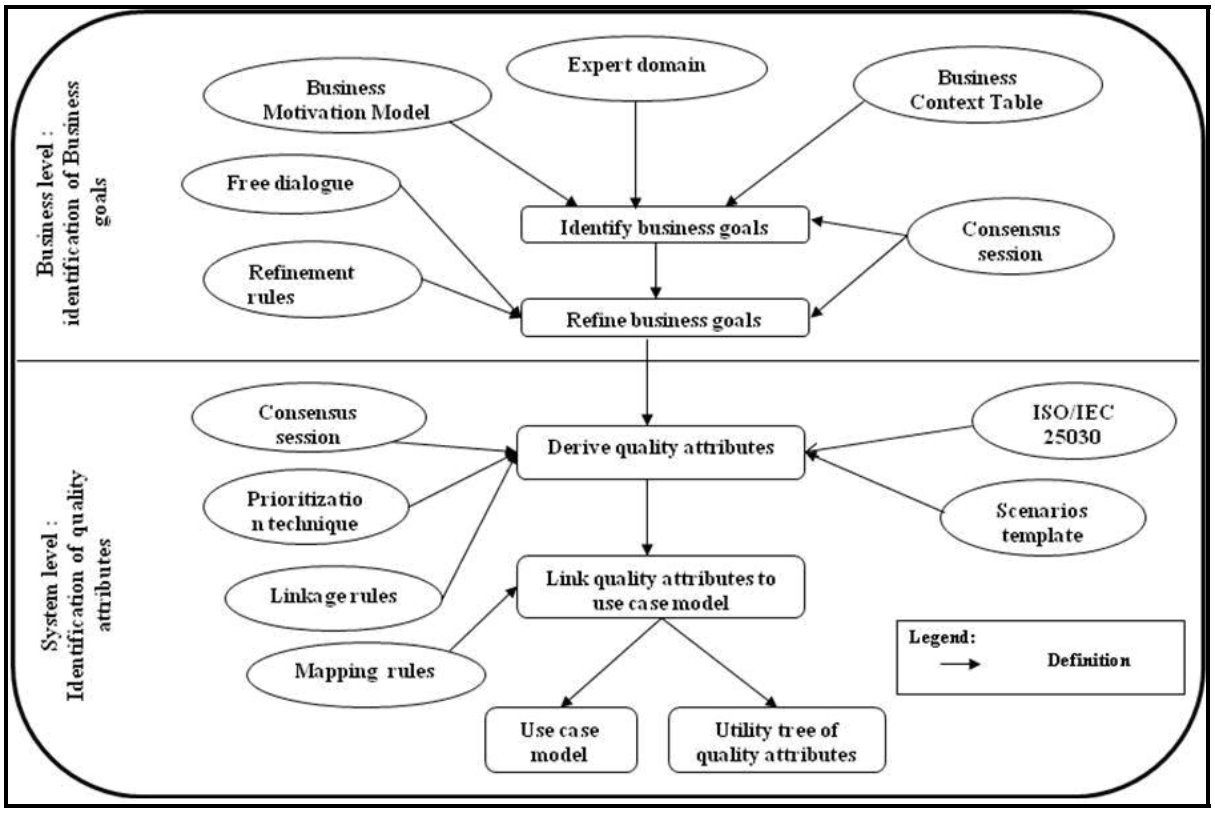


Figure 4.1 High conceptual levels of SOQUAREM

Figure 4.2 shows the required elements for identifying QAs. BCT elements (which are questions on business context: What, Why, How and Who) are mapped with BMM artifacts to refine the business goals (**BGk**). Refined business goals are linked to QAs (according to quality standard ISO/IEC 25030, linkage rules, scenarios template and prioritization techniques) to obtain the final quality attributes list (**QAm**). QAs list is discussed with concerned stakeholders during consensus sessions.

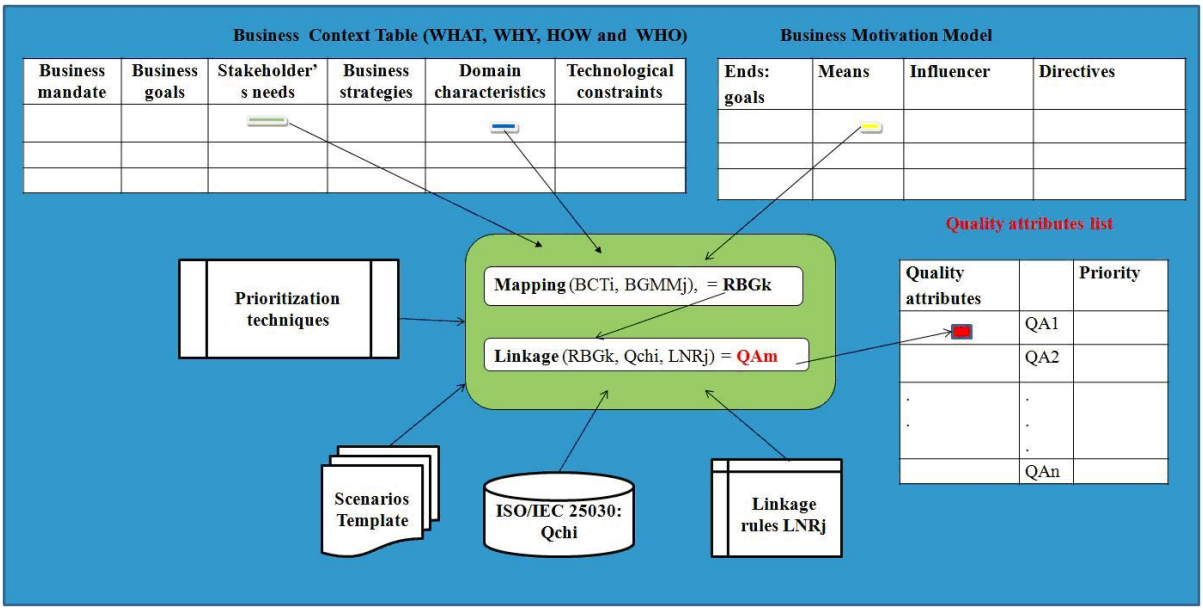


Figure 4.2 Required elements for identifying quality attributes

Data collected from the different questionnaires will produce the first database which will be organized and aggregated according to stakeholder's quality needs, ISO/IEC 25030 quality standard, scenarios template, linkage rules and prioritization techniques to finally obtain a list of prioritized quality attributes (Figure 4.3).

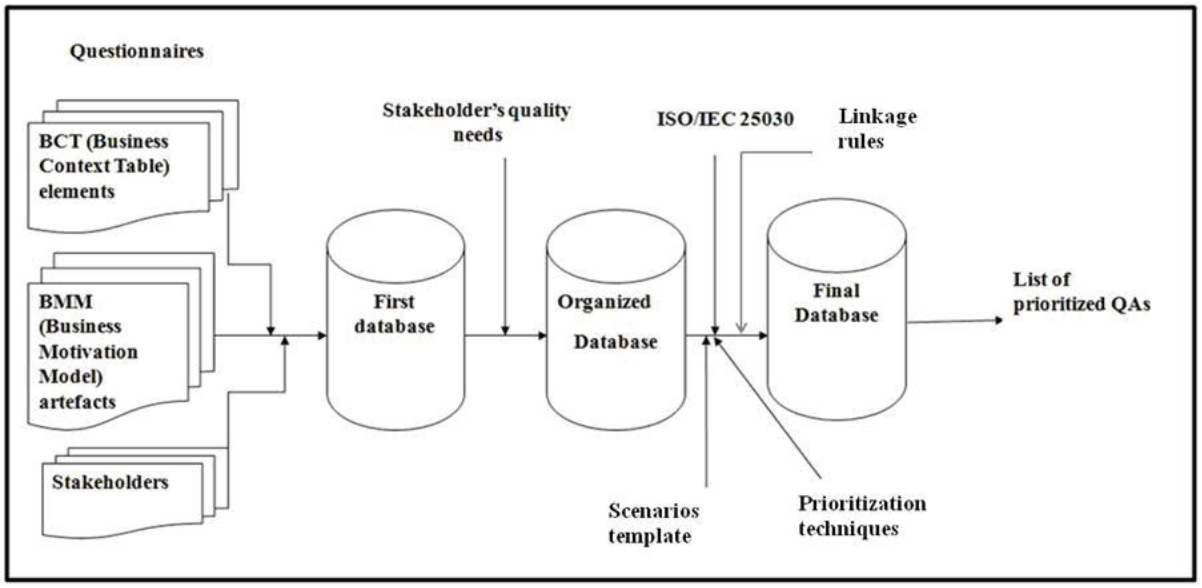


Figure 4.3 Process producing the quality attributes list

4.2 SOQUAREM Key concepts

This section describes the relevant concepts of SOQUAREM. The first group of concepts is provided from standards (BMM and ISO/IEC 25030) and known methods like priorizing methods (impact matrix and wheigted method), consenssus and free dialogue sessions. The second group of concepts is reused in SOQUAREM with Djouab's definitions. For instance, the "Utility tree" concept of ATAM (Kazman et al., 2000) method is reused with Djouab's representation model and the BCT concept is reused from the work of Deng (Deng, 2006) and QAs description template is reused from the work of Brito (Brito et al., 2003). The last group of concepts is developed for the purpose of SOQUAREM process (scenario template and transformation rules (statement, refinement, linkage and mapping rules). The Key concepts of SOQUAREM (Figure 4.4) are:

1. **BMM (Business Motivation Model):** is the starting point of the SOQUAREM method. It is used to define motivation of the business context, state goals and sub goals of the business, related strategies and identifies relevant stakeholders with their corresponding expectations.
2. **Business context Table (BCT):** describes fundamental questions about elements of the business context. It structures and details items of BMM business context according to the following keywords questions: How, What, Why and Who. BMM and BCT are used in the first three SOQUAREM process phases to help refine business goals and derive quality attributes from business goals.
3. **Free dialogue session**: is used to identify and refine business goals from technological constraints, high level functional requirements and covering strategies.
4. **Scenario template:** details the meanings of quality attributes according to specific items of the scenario template (Table 4.11). The scenario template provides a structured way to build the QAs utility tree and to integrate QAs in the FRs model (Use cases).
5. **ISO/IEC 25030:** helps stakeholders focus on the most recognized quality characteristics. It is used to infer the right quality attribute from the refined business goals.

6. **Consensus session:** provides a means to communicate and consolidate quality attributes to the stakeholders in order to obtain the final list of prioritized quality attributes. Consensus sessions are used to :
   a. Confirm business goals with stakeholders;
   b. Discuss the linkage of the QAs to the business goals with concerned stakeholders;
   c. Confirm consolidated QAs.
   d. Discuss conflicts among QAs with stakeholders.
7. **Quality attributes template:** documents quality attributes in the following terms: the context in which the quality attribute is applied, the source of the quality attribute, representation of the quality attribute and impact of the quality attribute on the software process.
8. **Utility tree:** (for traceability of quality attributes) is developed for each quality attribute and shows how quality attributes are organized with the refined business goals and the associated quality scenarios.
9. **Statement, refinement, linkage and mapping rules:** state and define ways to refine business goals, link quality attributes to the refined business goals and map quality attributes to the corresponding use case model.
10. **Prioritizing methods:** (such as impact matrix and weighted method) used to find and resolve conflicts among quality attributes.

Figure 4.4 presents key concepts involved in the main activities of SOQUAREM process. The first activity related to identifying and refining business goals (green color) uses the following concepts: BMM, BCT, free dialogue session, consensus session, statement and refinement rules. The second activity addresses derivation of quality attributes from the refined business goals and their consolidation by applying the following concepts: BMM and BCT, scenarios template, quality standard ISOIEC 25030, linkage rules, consensus session and prioritizing techniques (yellow color). The next activity uses the “Mapping rules” to link QAs to the use case model (blue color). The last two activities apply “QAs template” and “Utility tree” concepts to deal with documentation and representation of quality attributes (red and purple colors).

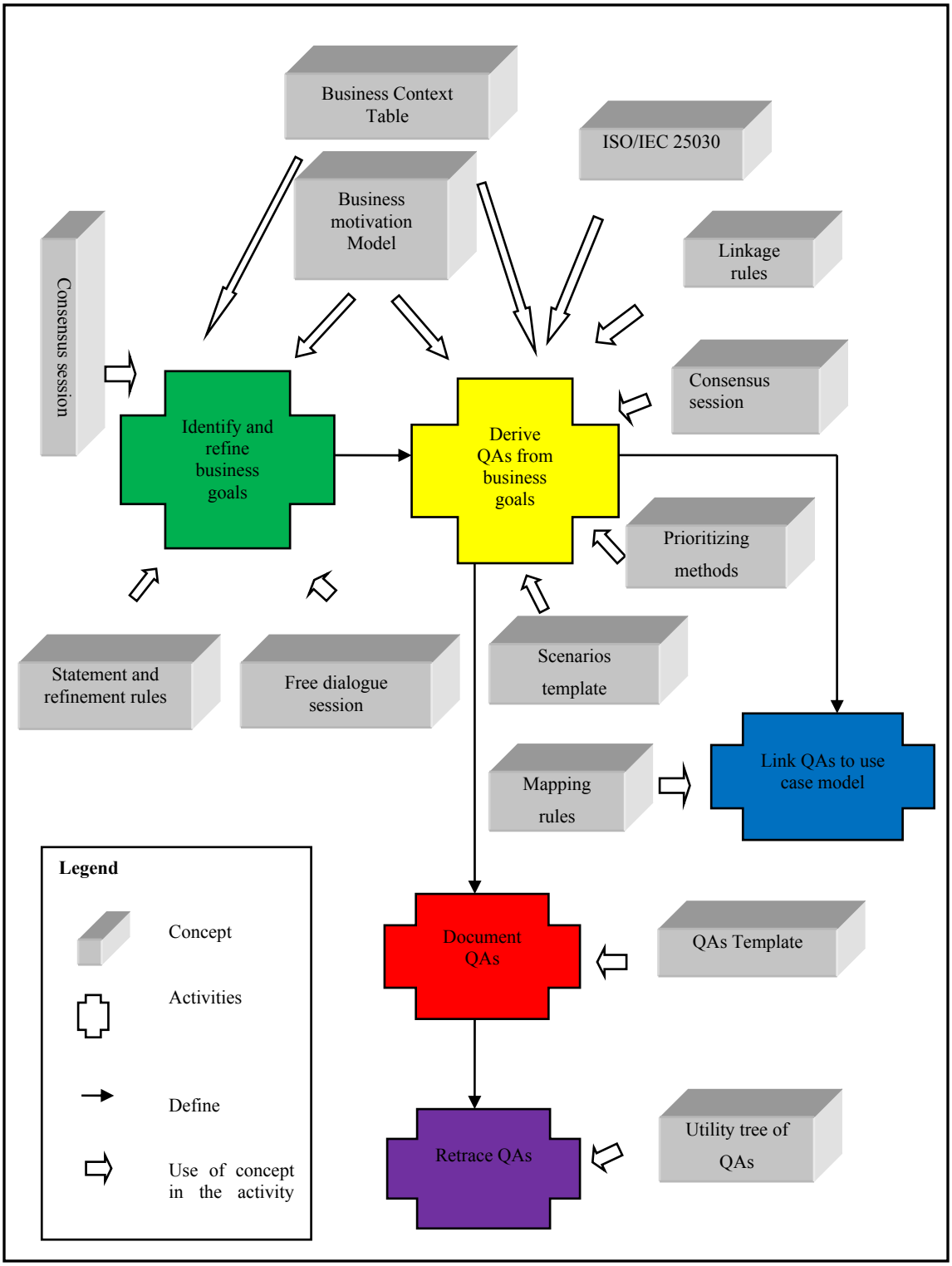


Figure 4.4 Key concepts of SOQUAREM

4.2.1 Development of SOQUAREM concepts

This section details the main concepts developed in SOQUAREM method: the BCT (Business context Table), scenario template, utility tree, QAs template, consensus session and the different transformation rules used in SOQUAREM process (statement, refinement, linkage and mapping rules). The BMM model is detailed in chapter 1 (section 1.4.12).

4.2.1.1 The BCT

The idea of BCT is to structure the business vision of the system by using keyword questions such as: what, why, who, where and when. BCT will help to organize business information that defines the scope of SOQUAREM process.

As suggested by its name, a business context is organized around questions related to identification and clarification of business context elements which contribute to identify quality attributes. Table 4.1 presents the elements of a business context.

Table 4.1 BCT (Business Context Table)

Questions	Business context elements
What	1. Business goals 2. High level and technological constraints 3. High level quality needs 4. High level functional requirements 5. Regulations and compliance 6. Domain characteristics 7. Political interests and organizational culture
How	Business strategies to achieve business goals
Who	Target stakeholders
Why	1. Current business a. Outcomes b. Impact c. Performance measures 2. Needs for target stakeholders to be met 3. Business mandate

4.2.1.2 The consensus session

The consensus session is used to discuss and consolidate ideas with stakeholders about quality attributes. It is applied throughout the entire process of SOQUAREM by using different techniques (C/R is used to indicate Confirmed/Rejected):

1. During the first two phases, a consensus session is used to confirm and to consolidate the business goals and refined business goals with stakeholders;
2. During phase 3, this session is used to confirm linkage of quality attributes to the business goals with stakeholders (Table 4.2);
3. Phase 5 uses the consensus session combined with the weighted quality attributes method to help resolve conflicts among quality attributes and discuss them with stakeholders (Table 4.3);
4. Phases 4 and 6 also use a consensus session technique to confirm with stakeholders the obtained quality scenarios and to map the QAs to the FRs model.

Table 4.2 Confirm linkage of QAs with business goals

Consensus session **Description:** Confirm linkage of quality attributes with business goals Stakeholders involved: Manager, developer			
Business goals	**Refined business goals**	**Assigned quality attribute to the refined business goal**	**Quality attributes** **Confirmed/ Rejected**

Table 4.3 Resolve conflicts among QAs

Consensus session **Description:** Resolve conflicts among quality attributes **Stakeholders involved:** Developer and evaluator			
Actor **Quality attribute**	**Actor1**	**Actor2** **.....**	**Actorm**
QA1			
QA2 **....**	**Weight [0...1]**		
QAn			

4.2.1.3 The QAs template

A template is used to document QAs in descriptive items which are subdivided into three classes (Tables 4.4 and 4.5):

- **Quality attributes context class:** contains items documenting:
  - Name of the quality attribute which is defined according to ISO/IEC SQuaRE 25030;
  - Brief description of the quality attribute;
  - Category of the QA according to ISO/IEC 25030;
  - Source of information contributing to the definition of the quality attribute (stakeholders and documents);
  - Stakeholders impacted by the quality attribute: which class of stakeholders is interested by this QA;
  - Priority of the QA: expresses the importance of the quality attribute for the stakeholders. It can be :
    - High (H:), Medium (M) and Low (L);
    - Or by values like: [0.6…1] for High,] 0.3…0.6[for Medium and [0…0.3] for Low.

- **Quality attributes traceability class:** contains items documenting traceability of the quality attribute to its original business goals source. The "Representation" item is described by:
  - The list of business goals and the refined goals which are contributing to the derivation of the quality attribute;
  - Actors responsible for achieving the quality attribute;
  - The number of identified QAs scenarios for each actor.

- **Quality attributes impact class:** contains items documenting:
  - Requirements affected by the quality attribute like functional requirements, cognitive requirements;
  - Models and processes requiring the quality attribute like sequence diagrams, use case model and business domain model (Table 4.6);
  - Activities of the software life cycle and phases of the software process standards where this quality attribute is required, managed or verified (Table 4.7);
  - Impact of other quality attributes (negatively or positively) on the quality attribute (Table 4.8).

Table 4.4 Template for specifying quality attributes

<table>
<tr><th colspan="5">QA context class</th></tr>
<tr><td>Name</td><td colspan="4">The QA name</td></tr>
<tr><td>Description</td><td colspan="4">A brief description of the quality attribute</td></tr>
<tr><td>Category</td><td colspan="4">The QA category according to the ISO/IEC 25030 taxonomy : QiU, EQ and IQ</td></tr>
<tr><td>Source</td><td colspan="4">Stakeholders, vision document , use case artifacts</td></tr>
<tr><td>Target stakeholders</td><td colspan="4">Manager, customer, developer, quality evaluator and other stakeholders</td></tr>
<tr><td>Quality standard used</td><td colspan="4">ISO/IEC SQuaRE 25030</td></tr>
<tr><td>Priority</td><td colspan="4">Priority of the QA</td></tr>
<tr><th colspan="5">QA traceability class</th></tr>
<tr><td rowspan="2">Representation</td><td rowspan="2">Business goals</td><td rowspan="2">Refined business goals</td><td>Actor 1</td><td>Actor i</td></tr>
<tr><td>Number of quality scenarios</td><td>Number of quality scenarios</td></tr>
<tr><th colspan="5">QA impact class</th></tr>
<tr><td>Requirements</td><td colspan="4">Functional, non functional, cognitive and other type of requirements</td></tr>
<tr><td>Activities and phases</td><td colspan="4">Requirements elicitation, requirements analysis, architectural design and test.</td></tr>
<tr><td>Models and processes</td><td colspan="4">Sequence diagrams, uses case diagrams and architectural styles.</td></tr>
<tr><td>Impact</td><td colspan="4">Represents how a quality attribute can be affected by other quality attributes. This impact can be positive (+) or negative (-).</td></tr>
</table>

Table 4.5 QAs documentation classes types

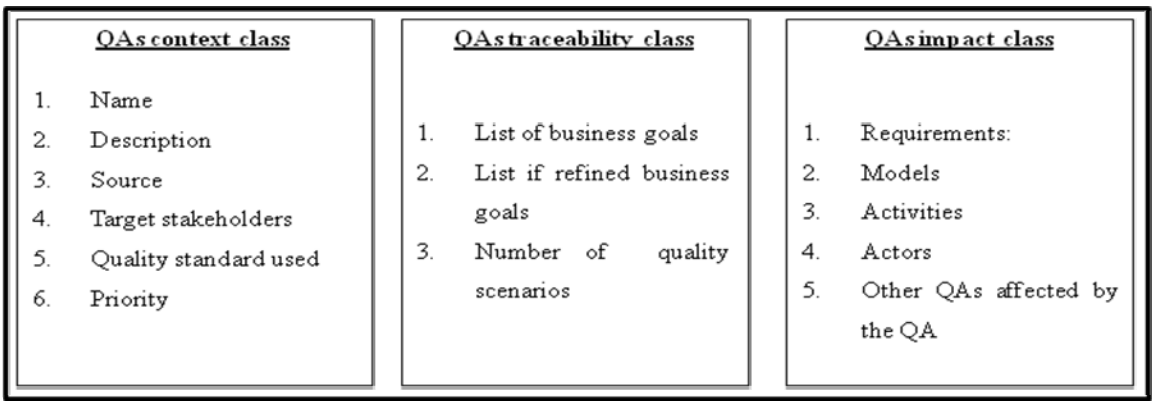


Table 4.6 Models requiring the QA

Activity / Where QAs	Use case model	Sequence diagrams	Architectural styles
Is required	*		
Is managed		*	
Is verified for its realization			*

Table 4.7 Activities requiring the QA

Activity / Where QAs	Requirements elicitation	Requirements analysis	Architecture design
Is required	*		
Is managed		*	
Is evaluated			*

Table 4.8 Impact matrix for conflicts among quality attributes

QAi / QAj	Quality attribute 1	Quality attribute 2	.....	Quality attribute n
Quality attribute 1	+	+		
Quality attribute 2				–
. . .				
Quality attribute n	+			

Figure 4.5 shows an overview of the QAs database representing the required data for the QAs management process. QAs are derived from the business goals, applied in a specific domain and could affect other requirements. They are described and detailed in scenarios where an action item contributes to make them operational. Actors are responsible for achieving the QAs and stakeholders are concerned with their realization. QA is required by different development models and is managed or verified in many activities of the software life cycle and phases of the software process standards.

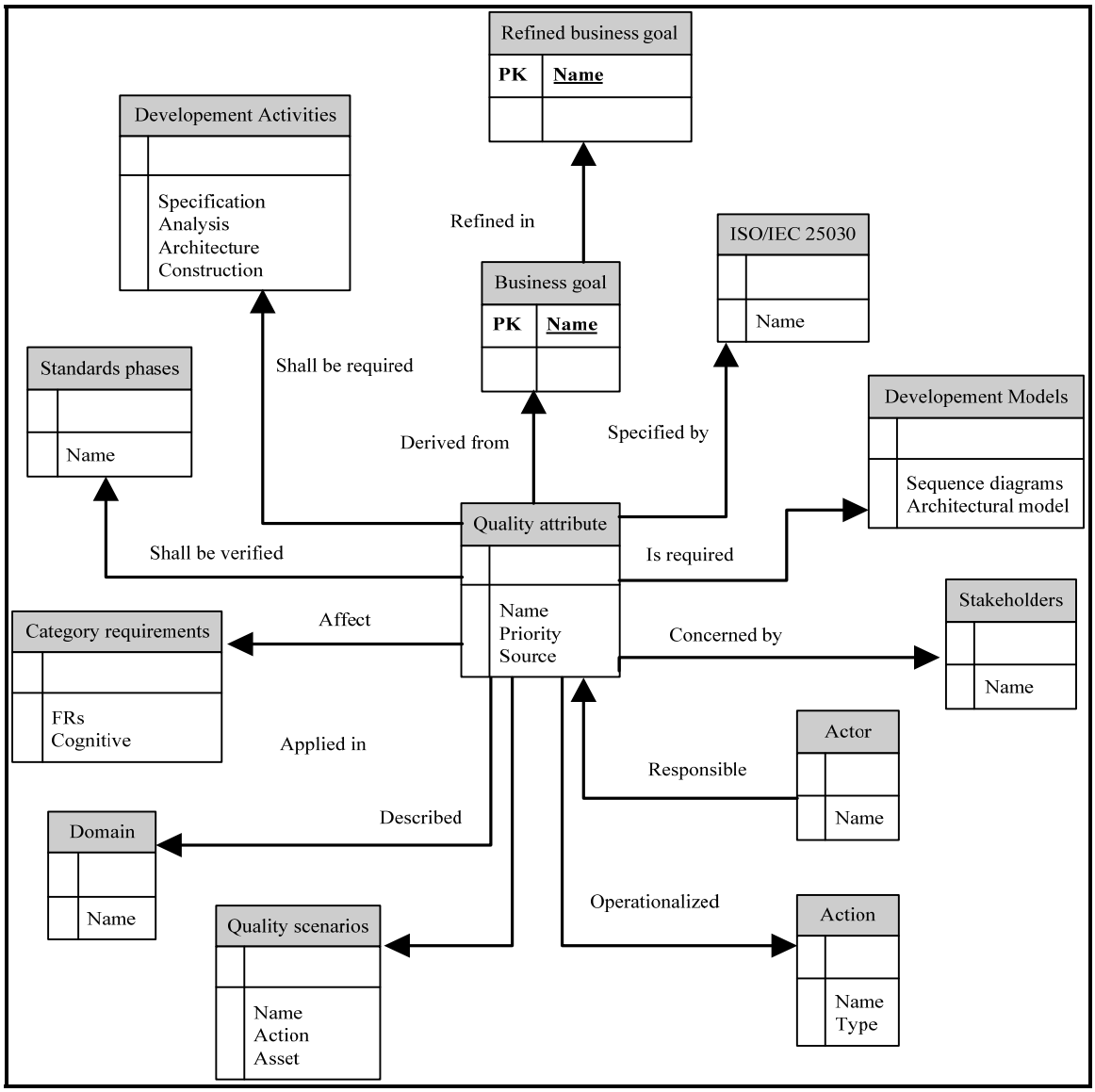


Figure 4.5 Quality attributes database reference

4.2.1.4 The scenarios template

The "Scenario template" provides a context for detailing and operationalising quality attributes. It is used to build the QAs scenarios and to map them to the FRs. The "Action" item is used to perform the mapping and the "Asset" item is used to elaborate the QAs scenarios. Table 4.9 summarizes the scenario description template items.

Table 4.9 Quality scenarios template

Scenarios items	Description
Action	Undertaken to achieve the quality attribute
Asset	Any part of the system (hardware, software, personnel, development process and data) involved in achieving the quality attribute

4.2.1.5 The utility tree

The utility tree is a key concept of SOQUAREM method. It is developed to describe the traceability of the quality attribute to its original requirements source. It represents derived quality attributes, their refined business goals and generated scenarios in a goal graph structure (Figure 4.6). It is structured into three levels:

1. **Business level:** where stated business goals and their refined business goals are represented. Priority of the related refined business goal is also represented.
2. **Quality attributes system level:** where derived quality attributes are represented from detailed business goals. The actor responsible for achieving the quality attribute is also represented at this level.
3. **Scenarios System level:** where the meaning of the derived quality attribute is detailed with scenarios according to the scenario template (Table 4.9).

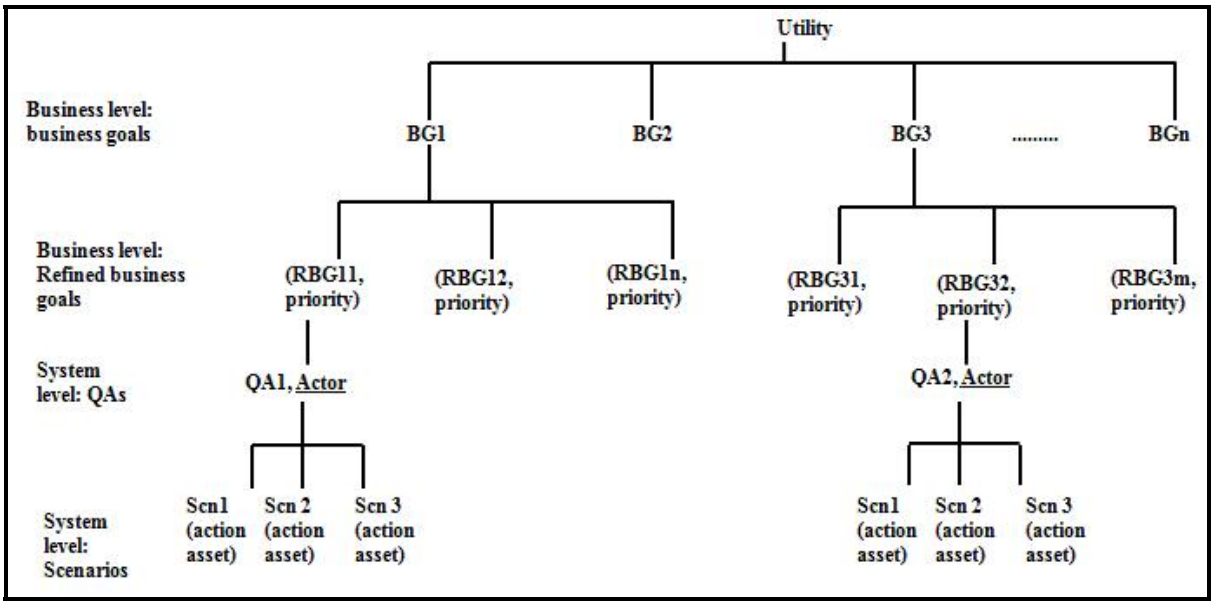


Figure 4.6 Utility tree of quality attributes

4.2.1.6 Transformation rules

This section describes the transformation rules used to verify the logic of SOQUAREM process. The proposed rules are applied at each phase of the process to help derive quality attributes. They are divided into the following rules:

1. **Statement rules:** used to ensure that business goals are stated according to a business mandate, domain characteristic and organizational culture of the business (Table 4.1). Table 4.10 gives an excerpt of statement rules in the SOQUAREM concept.

Table 4.10 Statement rules

Statement rules
1. **STR1 :** Each business goal is detailed according to "Business mandate" of BCT table and "Desired results" of BMM model; 2. **STR2:** Each business goal is related to one or more quality needs of stakeholders; 3. **STR3:** Each business goal is identified according to domain characteristics of the business; 4. **STR4:** Each business goal is defined according to high level problems and technological constraints of the business; 5. **STR5:** Business goals are defined according to regulations and compliance, political interests and organizational culture of the business.

2. **Refinement rules:** used to ensure that business goals are detailed according to covered business strategies, regulations, technological constraints and the organizational culture of the business. Table 4.11 gives an excerpt of refinement rules in the SOQUAREM concept.

Table 4.11 Refinement rules

Refinement rules
1. **RFR1:** Each business goal is detailed according to technological constraints, existing regulations and compliance and high level functional requirements; 2. **RFR 2:** Each business goal is detailed according to definition of the business strategies suggested to achieve the business goals; 3. **RFR3:** Business strategies of BCT should correspond or be part of courses of actions of BMM model.

3. **Linkage rules:** used to ensure that quality attributes are derived from refined business goals according to stakeholder's quality needs and ISO/IEC 25030 quality standard. Table 4.12 gives an excerpt of linkage rules in the SOQUAREM concept.

Table 4.12 Linkage rules

Linkage rules
1. **LNR1:** Each quality attribute is derived according to high level quality needs, definition of the refined business goal and taxonomy of ISO/IEC 25030;
2. **LNR2:** Each derived quality attribute could be linked to one or more refined business goal;
3. **LNR3** Each obtained quality attribute could be achieved by at least one actor;
4. **LNR4:** Define relevant actors who should achieve quality attribute from **e**xternal influencer of the BMM model;
5. **LNR5:** the WHO item: target stakeholders of the BCT should be part of external influencer of BMM model;
6. **LNR6:** Define relevant actions from definition of refined business goals and internal influencer of the BMM model;

4. **Mapping rules:** used to ensure that quality attributes are mapped to functional requirements (the use case model) by using scenario template items as main drivers of this mapping. Table 4.13 gives an excerpt of mapping rules in the SOQUAREM concept.

Table 4.13 Mapping rules

Mapping rules
1. **MPR1**: Actor of the "Utility tree" is mapped to Actor of the use case model;
2. **MPR2:** "Action" of the QA scenario undertaken by Actor is mapped to a new use case;
3. **MPR3:** Each actor of the "Utility tree" is mapped to a business concept of the business domain model;
4. **MPR4: "**Asset" and "Action" of the QA scenario are mapped to business concept and relationship between mapped business concepts.

4.3 The SOQUAREM process model

The SOQUAREM process model is divided into six phases for defining and refining business goals, deriving, operationalizing, analyzing, documenting and representing QAs and finally for linking them to the FRs process. These phases use various software QRs management techniques (questionnaire, consensus session, BMM, scenarios, prioritizing, utility tree and template). Potential inputs to the process are BMM, BCT and domain experts. The main participants are quality requirements engineers, domain experts and selected stakeholders. If the outputs for each phase are not approved by the stakeholders, one can suppose that the stakeholders need to negotiate with each other during consensus sessions and the phase is restarted if necessary. The negotiation techniques are not investigated in this thesis.

The SOQUAREM process (Figure 4.7) is represented as:

Phase 1: State and identify the business goals: define the relevant elements of the business context such as business goals and business domain. It is important to mention that the business goals definition is related to the goals of the BMM concept.

Phase 2: Refine business goals: business goals are detailed according to additional business information such as organizational culture, regulations and guidelines, technological constraints and business strategies.

Phase 3: Link business goals to the corresponding quality attributes: detailed business goals are used to derive the quality attributes by using ISO/IEC 25030 quality standard and linkage rules. The relation between the business goals and the FRs is not included in this process.

Phase 4: Build quality attributes scenarios by using the scenario template and the consensus session techniques to infer the right quality attribute.

Phase 5: Analyze conflicts between QAs and consolidate them by using prioritization methods. If the consolidation is not approved by the stakeholders, the process is restarted from the phase 3 (Figure 4.7).

Phase 6: The last phase of one iteration cycle consists of linking the QAs to the functional requirements process by updating the initial use case model with additional information about QAs.

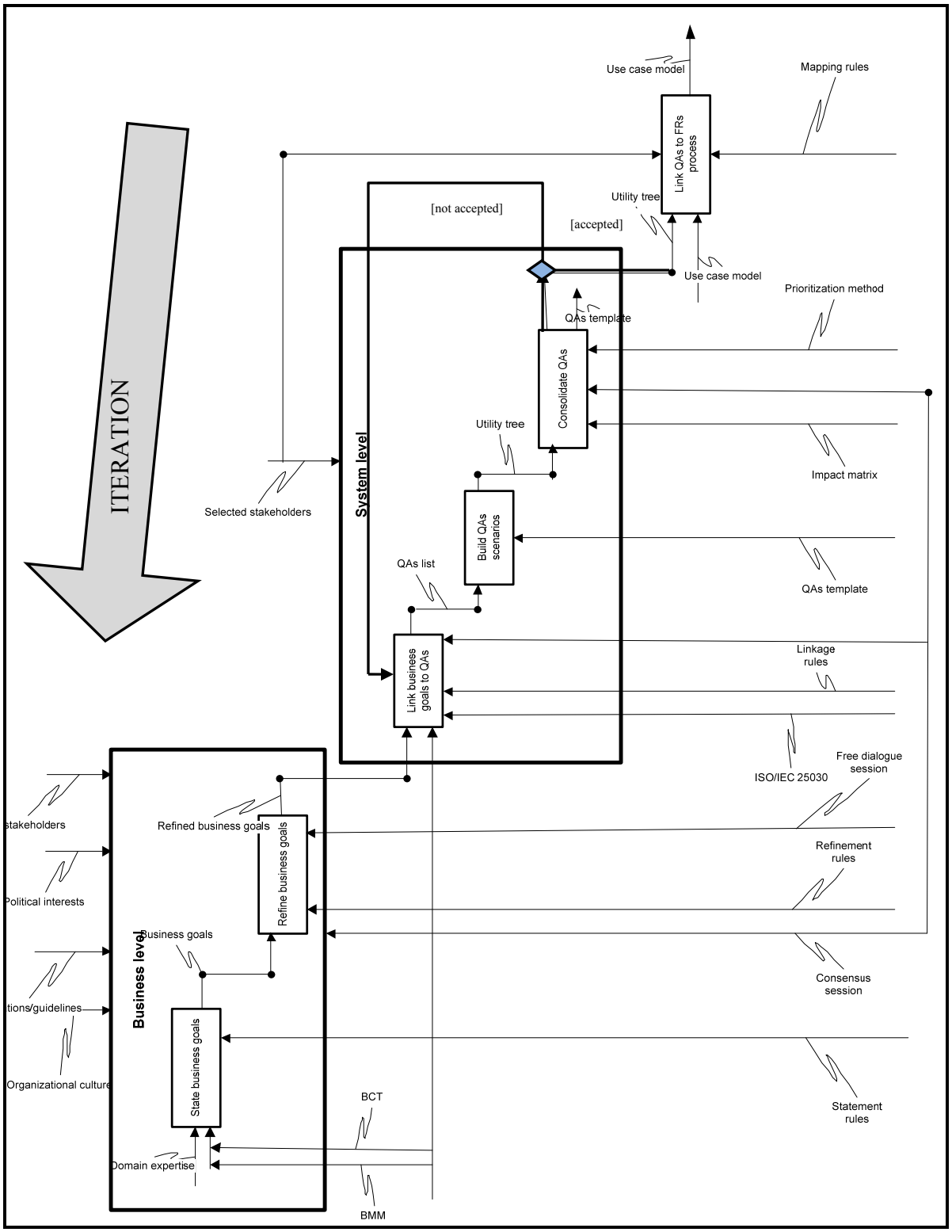


Figure 4.7 SOQUAREM process model

Figures 4.8 and 4.9 summarize the linkage process and logic of SOQUAREM. Figure 4.8 shows the linkage process of SOQUAREM involving elements of the business context (like business vision, business goals and strategies) to be refined and linked to system elements like quality attributes, actors and associated actions.

As illustrated by Figure 4.9, quality attributes are identified from business goals and integrated into the FRs process. SOQUAREM process is used at two levels: the business level where elements of the business context as BMM and BCT are used with the statement and refinement rules to help identify business goals and refine them into refined business goals. At the system level quality attributes are: a) linked to refined business goals by using quality standard ISO/IEC 25030 and linkage rules and detailed into quality scenarios; and b) mapped to the FRs process by using mapping rules and a scenario template. SOQUAREM helps to provide traceability of QAs to their business goals. Elements of the business context could be mapped to the QAs by:

1. Refining business goals into sub goals and linking them to quality attributes (blue, green and purple colors);
2. Deriving actors responsible for achieving quality attributes from the "External influencer" item of BMM (Deng, 2006) and the "Who" questions of BCT concepts (red color);
3. Deriving actions undertaken by actors to achieve quality attributes from the "Internal influencer "item of BMM and the refined business goals (brown color).

The mapping from BMM elements like "Internal and external influencer" to the defined actors an actions is inspired from the work of Deng which suggests in her research to integrate the modeling techniques of BMM and I* framework.

Deng said that "An external influencer in the BMM could be considered as an actor in i*, and an internal influencer could be a resource, task, goal, softgoal, or belief according to its characteristics". She also said that to determine if they mean the same concepts, it will depend on further definitions of these concepts by OMG.

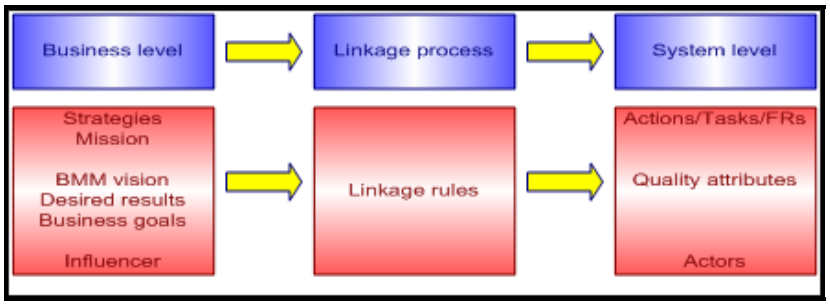


Figure 4.8 Linkage process of SOQUAREM process

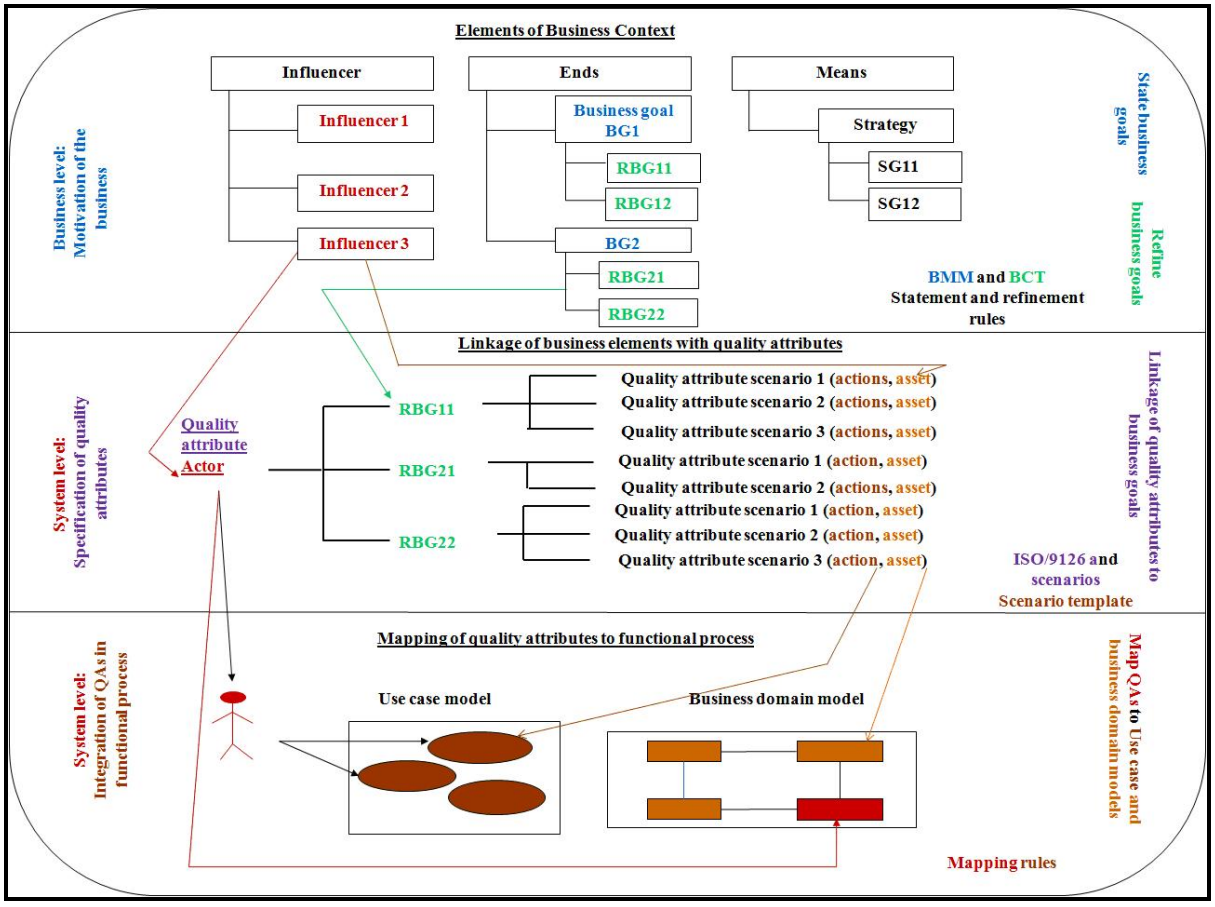


Figure 4.9 Logic of SOQUAREM process model

4.3.1 Detailed description of the phases of SOQUAREM process

Phase 1: State and identify business goals of the system (Figure 4.10)

In this phase, the business goals of the organization are formulated from the BMM and BCT items. The business goals definition starts from the goals of the BMM concept (the "Ends:: desired results::goals" item) (Table 1.10). The BCT concept provides the "WHY" (business mandate and target stakeholder's needs) and "WHAT" (high level problems, technological constraints, high quality needs, domain characteristics and organizational culture) artifacts to state the business goals. Statement rules and consensus sessions are techniques used to define and discuss business goals with stakeholders. Outputs of this phase are the main business goals of the system.

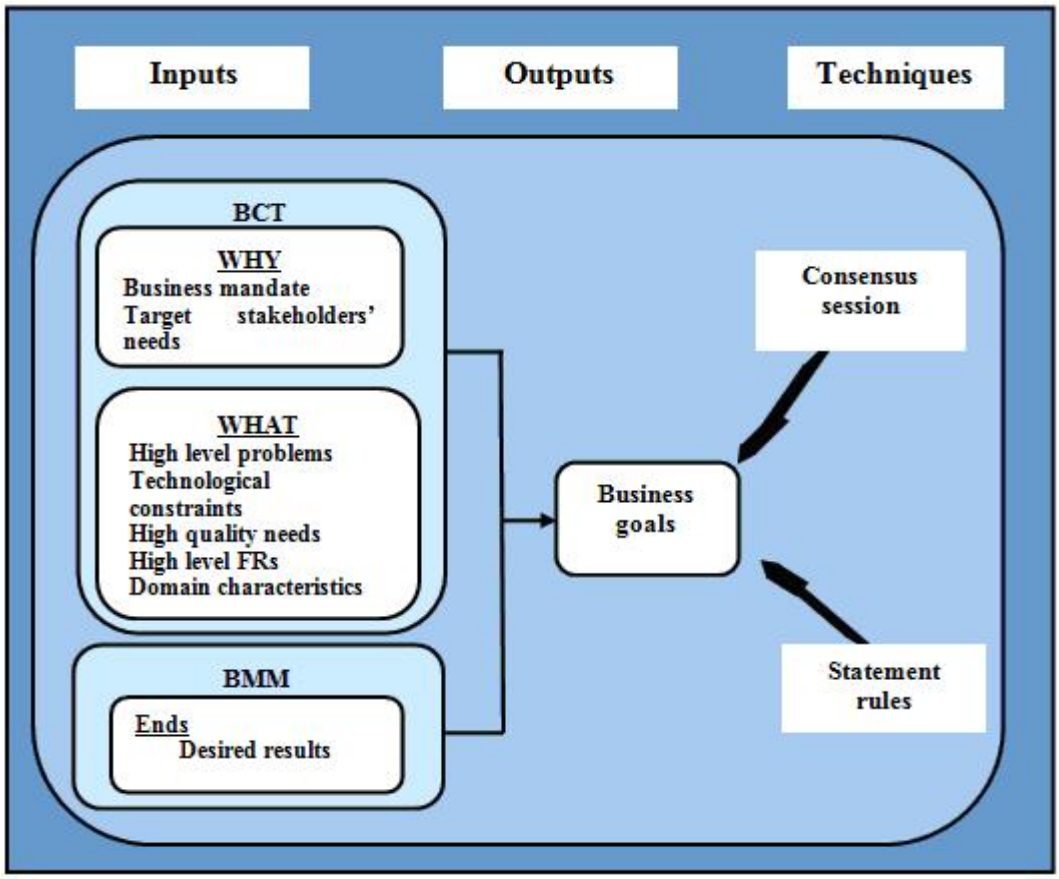


Figure 4.10 State the business goals

Phase 2: Refine the business goals (Figure 4.11)

This phase details the business goals with additional information by describing the covered strategies to achieve these goals, technological constraints, directives/regulations and organizational cultures impacting the business system. The BMM and BCT concepts are used as inputs in this phase. The BCT concept provides the following inputs: a) "HOW" (business strategies) and b) "WHAT" (high level functional requirements, technological constraints and regulations). The BMM concept provides the following inputs: a) "Course of action" (Strategies) and b) "Directives". Consensus sessions, free dialogue sessions and refinement rules are techniques used to discuss and confirm refined business goals with stakeholders. Refined business goals are prioritized with the participation of stakeholders as follows: High (H), medium (M) and Low (L). Refined business goals are the main output at this phase.

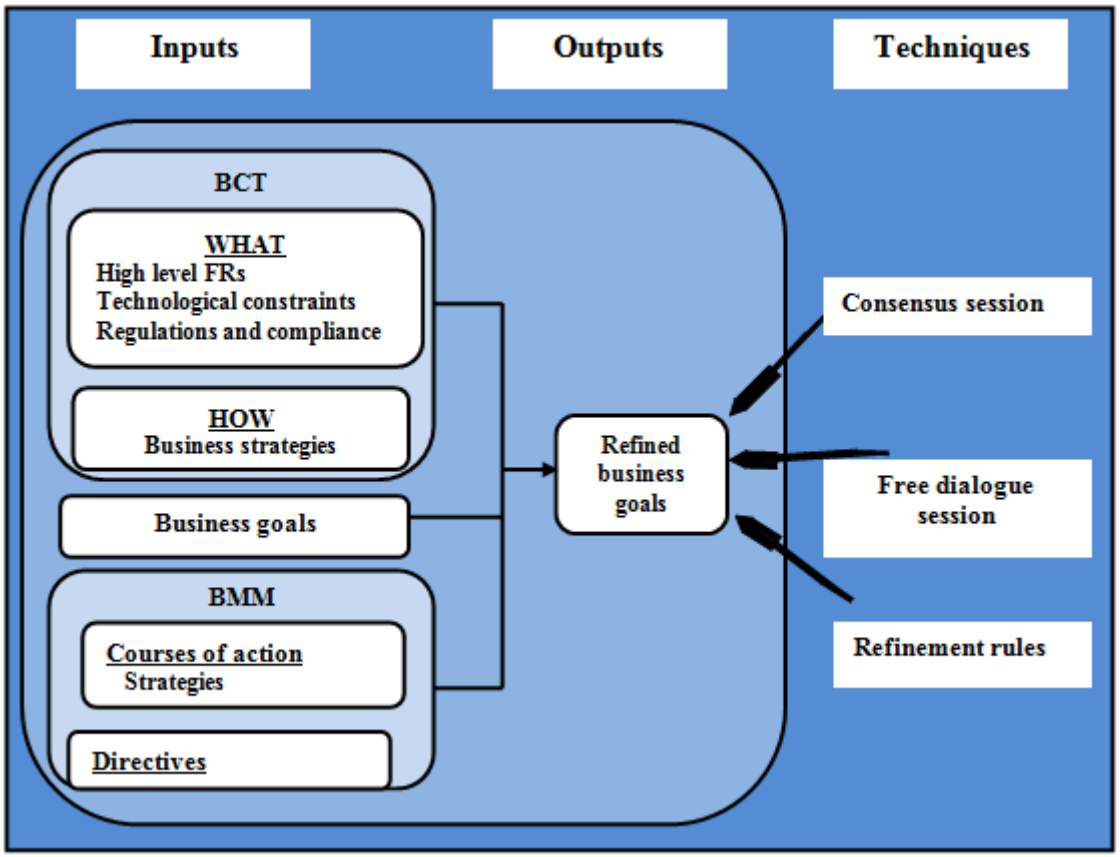


Figure 4.11 Refine the business goals

Phase 3: Link the business goals to corresponding quality attributes (Figure 4.12)

The phase derives the quality attributes and their associated actors and actions from the refined business goals by using linkage rules and ISO/IEC 25030. Relevant actors related to achievement of quality attributes are derived from the “WHO" question of the BCT concept (target stakeholders item) and the BMM concept (external influencer’ item). Actions are identified by asking questions about possible actions that could be derived from the “Internal influencer” item and the refined business goals. Quality attributes are derived from the “WHAT” question of the BCT concept (High level quality needs item) and ISO/IEC 25030 quality standard. Linkage rules are used to verify the derivation process of quality attributes. Consensus sessions are used to discuss and confirm obtained quality attributes with stakeholders. The output at this phase is a quality attributes list.

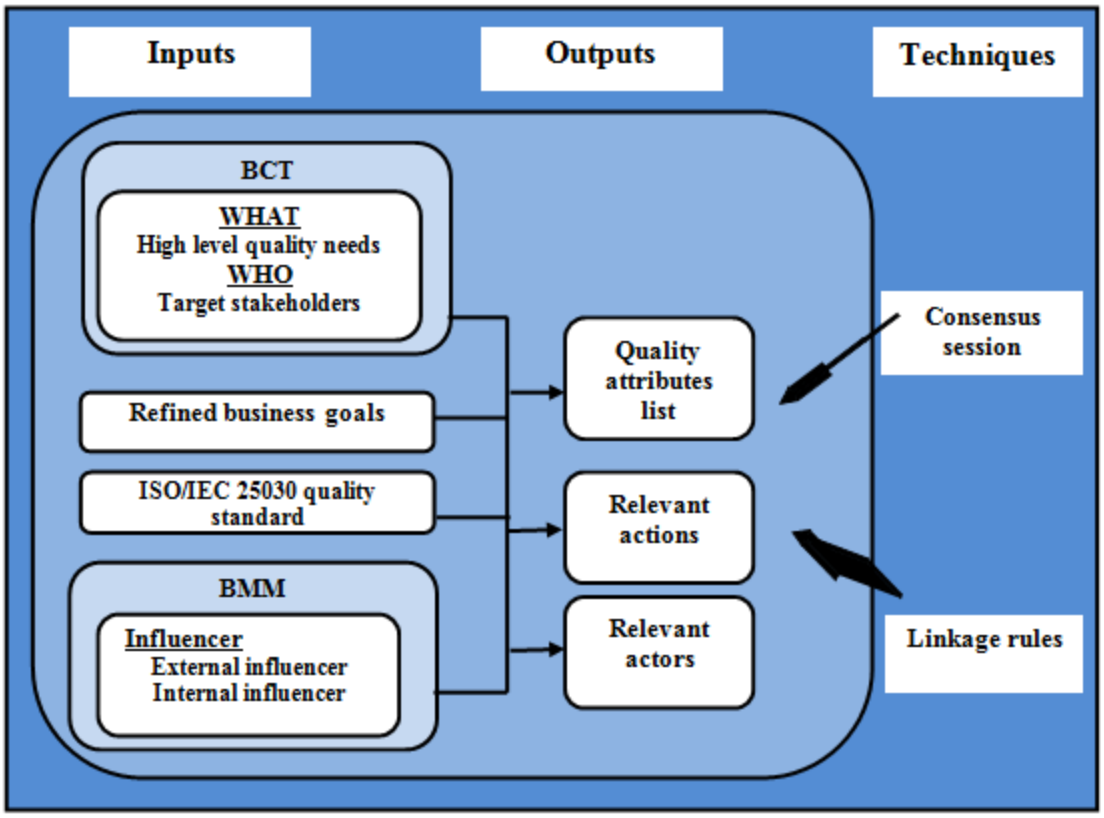


Figure 4.12 Link the business goals to the corresponding quality attributes

Phase 4: Build the quality attribute scenarios (Figure 4.13)

This phase builds the quality scenarios associated to the QAs according to the scenario template description (Table 4.9). Important items of the scenario template are: action undertaken to achieve quality attribute and asset on which action is undertaken. The Action item of the scenario template is mapped to the relevant action field of the QAs list. The asset item of the scenario template is defined from the refined business goals. Consensus sessions are used to confirm quality scenarios with stakeholders. The output at this phase is utility tree of quality attribute scenarios (Figure 4.6).

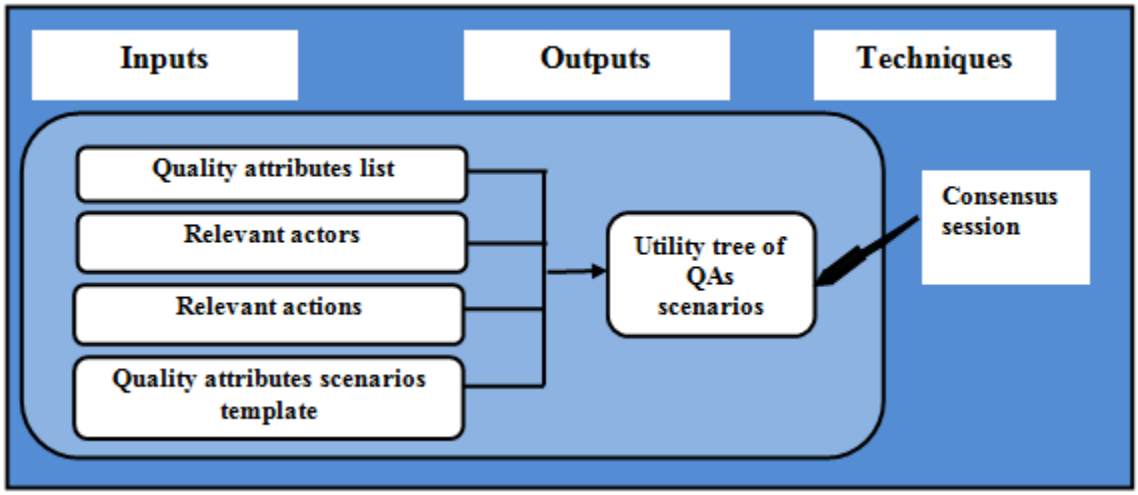


Figure 4.13 Build the quality scenarios

Phase 5: analyze conflicts among quality attributes and consolidate them (Figure 4.14)

This phase evaluates interactions among QAs, compares and adjusts them to find and remove conflicting QAs. It consists of:

- Building the impact matrix where each quality attribute may contribute negatively or positively to the other quality attributes in order to find possible conflicts and resolve them (Table 4.8).
- Attributing weights (range [0...1]) represents priority) to those quality attributes that contribute negatively to each other (Table 4.14). The weighted method describes the

extent to which a quality attribute may constrain an actor. The values are given (by involved stakeholders) according to the importance each quality attribute has for each actor. In the case where all the values are rated high, a voting system is performed or a maximum per stakeholder is defined by using the negotiation techniques. The scales used here are based on fuzzy logic and have the following meaning [Rashid and al., Brito and al., 2002]:

- *"Very important"* takes values in the interval [0,8 .. 1,0]
- *"Important"* takes values in the interval [0,5 .. 0,8]
- *" Medium"* takes values in the interval [0,3 .. 0,5]
- *"Low"* takes values in the interval [0,1 .. 0,3]
- *"Very low"* takes values in the interval [0 .. 0,1]

- Resolving conflicts and consolidating them with the stakeholders (during consensus session) by using the weighted method;
- Building the utility tree according to the consolidated data. Labels are assigned to each quality attribute scenario as follows:
  - S if the quality attribute is satisfied;
  - D if the quality attribute is not satisfied;
  - P if the quality attribute is partially satisfied.
- Repeating the process to select the most beneficial QAs which involve the least conflict.

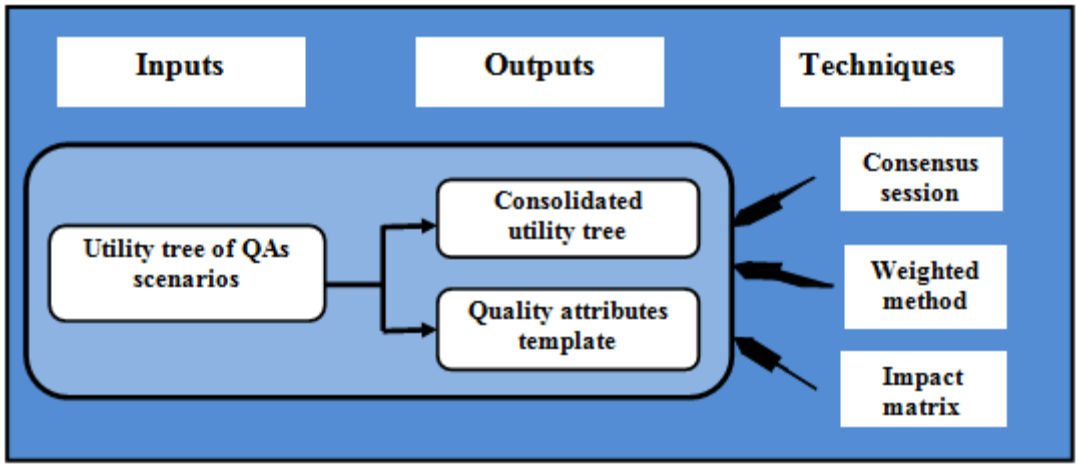


Figure 4.14 Analyze conflicts between QAs and consolidate them

Table 4.14 Attribute weights to quality attributes

Actor / Quality attribute / RGBi	Actor 1	.....	Actor m
QA1/RBG1			
QA2/RBG2	Weight [0…1]		
……			
QA1/RBGn			

Phase 6: link quality attributes to a functional model (use case model and business domain model) (Figure 4.15)

QAs are linked to the FRs process ("Use cases" and "Business domain" models) by using the mapping rules. These models are already defined in the beginning of the process and are enriched with QAs data. The following section describes the mapping rules of a QAs utility tree and "Use cases" and "Business domain" models.

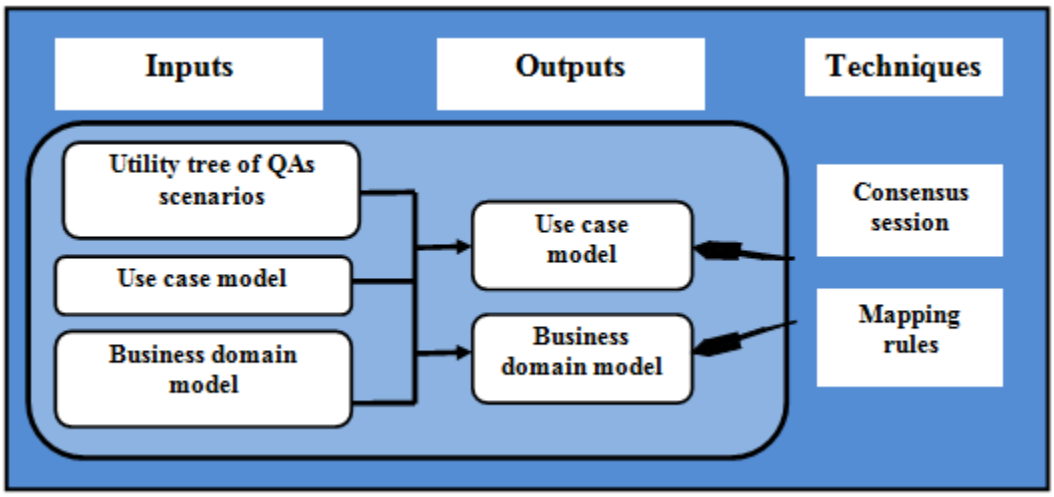


Figure 4.15 Link QAs to use case and business domain models

Figure 4.6 illustrates the mapping process of a use case model (with the actor "Actor1" and two actions "Action1" and "Action3") with two utility trees Utility1 and Utility2 in the following steps:

1. Find all the QAs utility trees that refer to a particular actor in the use case model (Utility 1 and Utility 2).
2. Relate actions of the actor to roots of the QA utility tree which correspond to the same actor (Action1 and Action3 of Actor1 are related to Utility1 and utility2).
3. If actions in the use case model do not cover the nodes of the QA utility tree (part actions), add the later actions of the actor to the use case model (Actions 2, 5 and 6).
4. If the actor of the QA utility tree does not exist in the use case model, add it to the use case model with its associated actions (Actor2 and Actor3).

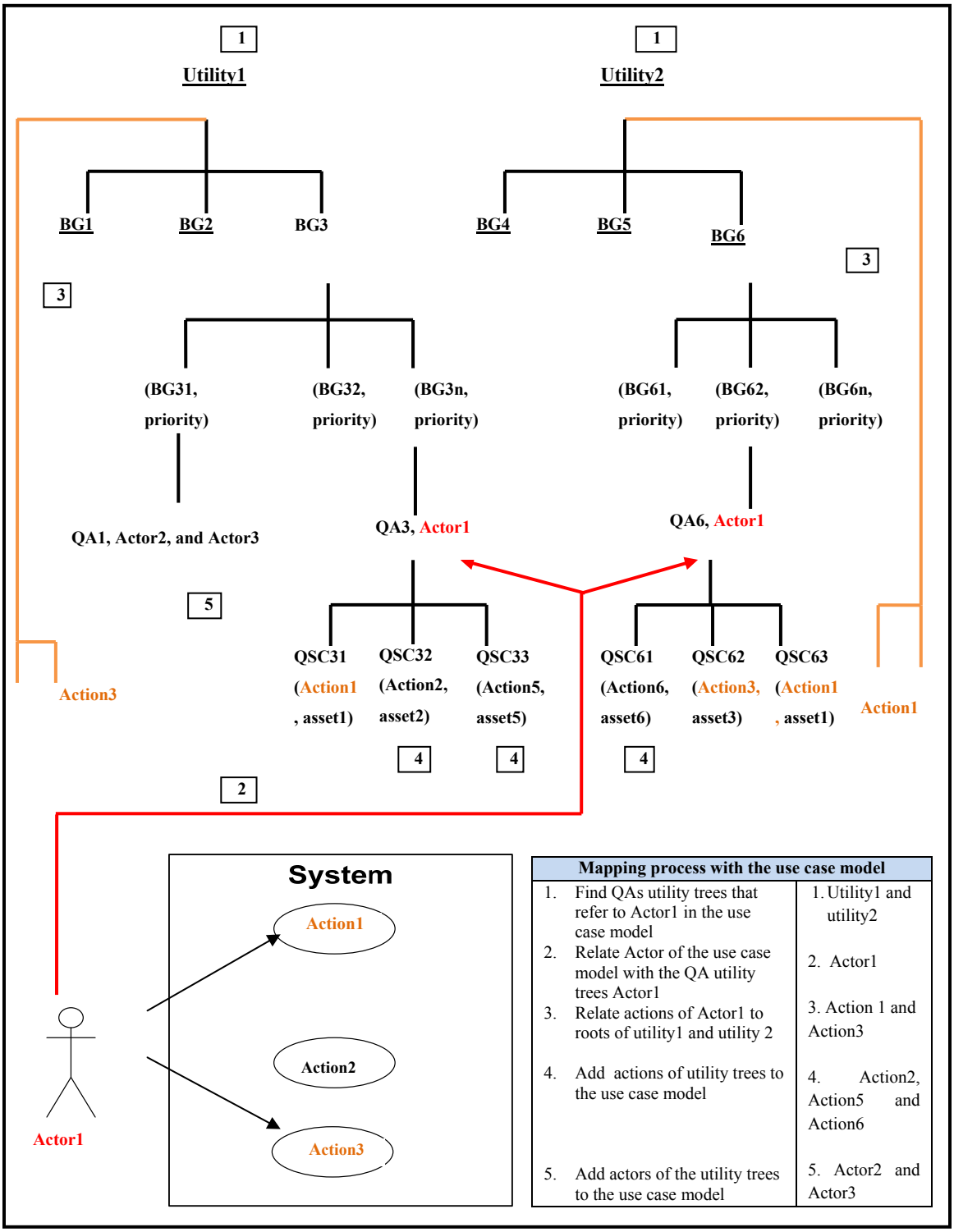


Figure 4.16 Mapping process with the use case model

Figure 4.17 shows the mapping process of a business domain model (containing the business concept: “Actor1” and the relationships “Relationship1” and “Relationship2”) with two utility trees Utility1 and Utility2 in the following steps:

1. Find all the QAs utility trees that refer to the business concept “Actor1” in the business domain model (Utility1 and Utility2).
2. Relate relationships of the business concept “Actor1” to roots of the QA utility tree which correspond to the same actor (“Relationship1” and “Relationship2”).
3. If relationships in the business domain model do not cover the nodes of the QA utility tree (actions and assets), add assets and actions of the actor (of the utility trees) to the business domain model as follows:

    i. Assets will be mapped to the business concepts and actions mapped to the relationship between actor concept and assets (Asset1, Asset2 and Asset3);
    ii. Actions will be mapped with more abstract relationships in the business domain model (Action1, Action2 and Action3). For example, action “add new language” in the utility tree (Figures. 5.21 and 5.24) will be mapped to “defines international language” relationship in the business domain model.
5. If actors of the QAs utility tree do not exist in the business domain model, add them to the business domain model with their associated “Action” and “Asset” nodes (Actor2 and Actor3).
6. Verify coherence and semantics of the extended business domain model (deleting all the redundant business concepts and update relationship with the same name).
7. QAs views are projected from the overall added business and relationship concepts (Figure 4.18) of the business domain model.

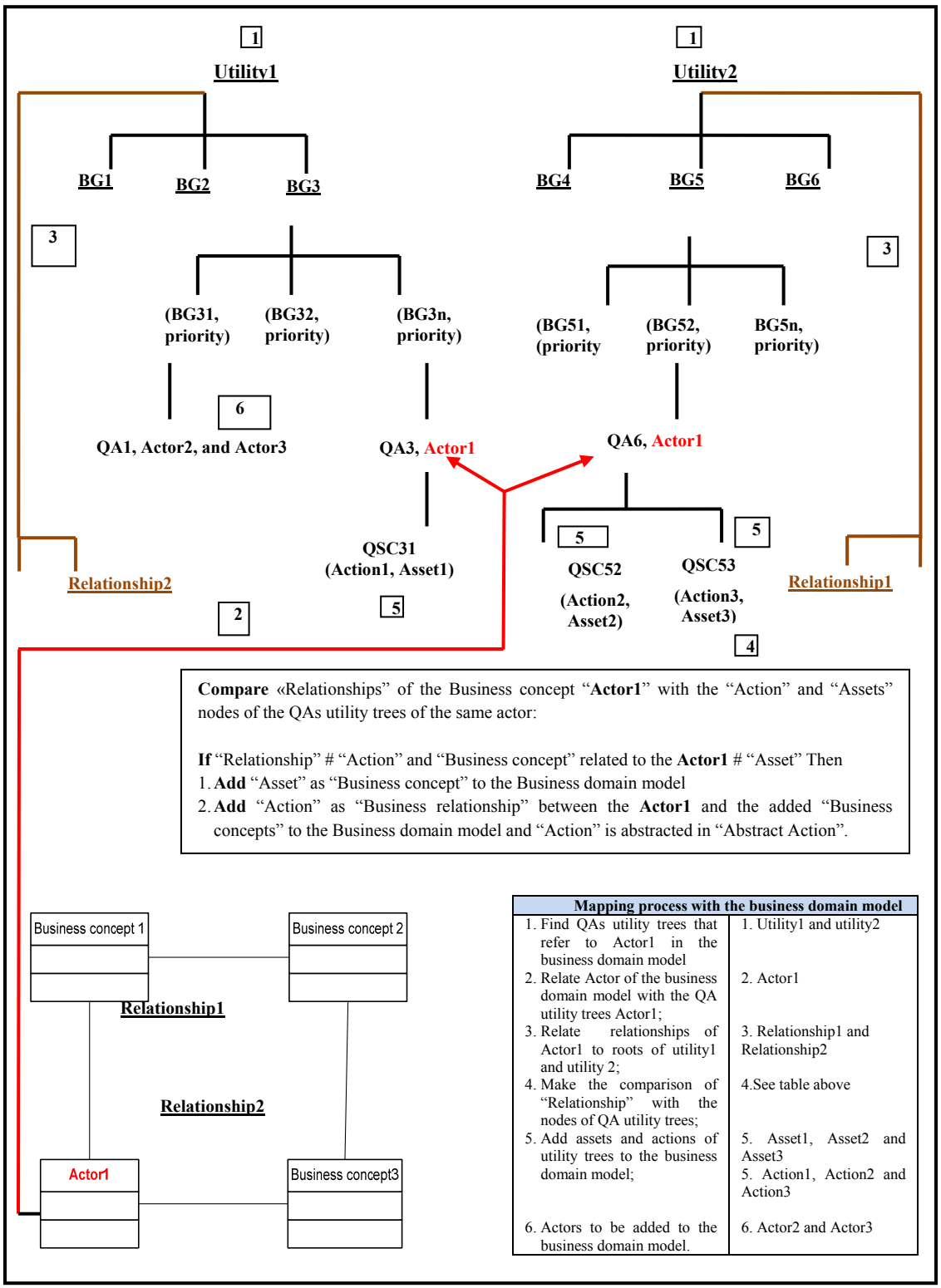


Figure 4.17 Mapping process with the business domain model

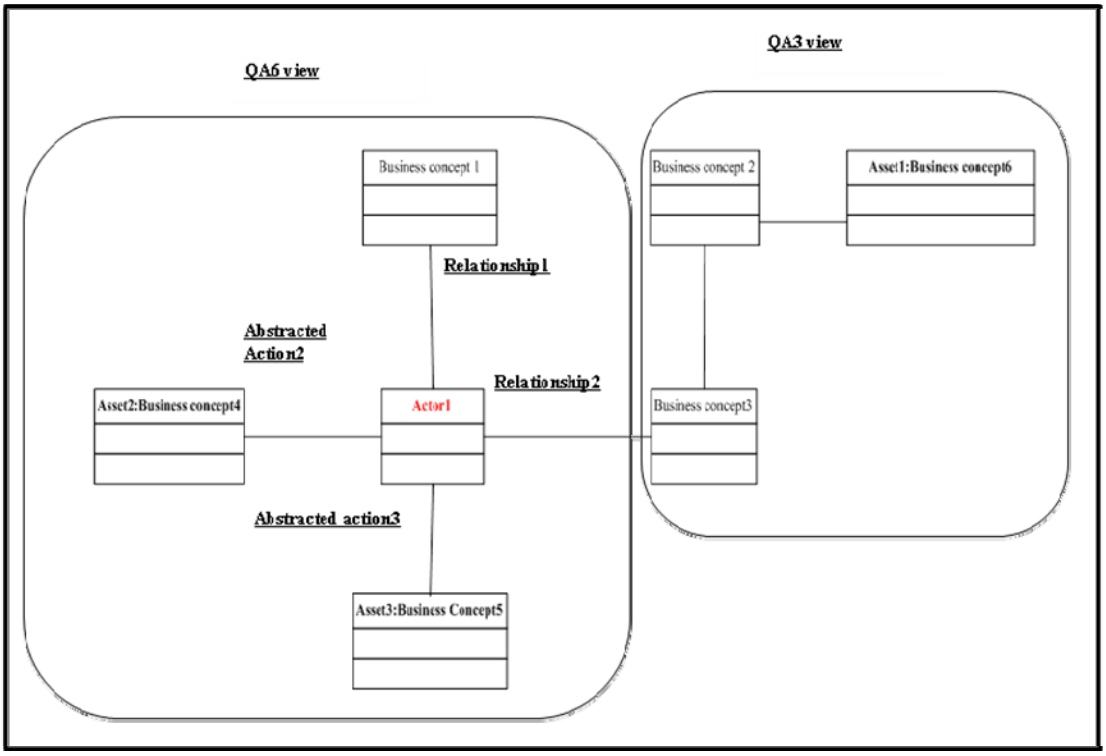


Figure 4.18 Quality attributes views

Quality attributes are derived from the business domain model. The new added business concepts (from actor of the utility tree and asset of the scenario template) and relationship (from action of the scenario template) help in the projection of the quality attribute view. Projection is specifically defined from relationship existing between the new added business concepts.

4.4 CONCLUSION

This chapter described the SOftware QUAlity Requirements Engineering Method (SOQUAREM) developed for quality requirements engineering process in the software product definition phase. SOQUAREM provides solutions to many recurring quality management problems which include:

- Systematic and structured identification, representation of QRs in the software product definition phase;
- Clear derivation of QAs from business concepts;
- Well defined traceability mechanism;
- Better integration of quality requirements with the functional process.

The main concepts of SOQUAREM have been described. One can cite:

- BMM (Business Motivation Model), BCT (Business Context Table) concept and transformation rules (statement, refinement and linkage) to identify and derive important QAs according to ISO/IEC 25030 taxonomy;
- Scenario template concept to infer the right QA and utility tree allows for describing the traceability of QAs to their original requirements;
- Prioritization methods (impact matrix and weighted method) help to analyze and resolve conflicts among QAs;
- QAs template to document QAs;
- Mapping rules and scenario template contribute to integrate QAs into the functional model;
- Finally, consensus sessions are used at each process phase to interact with stakeholders and domain experts.

Subsequently, SOQUAREM process phases are described in detail. They are structured as follows:

- The first 2 phases are used at the business level to identify and refine business goals;
- The last four phases are used at the system level to:
    - Derive QAs from the refined business goals;
    - Build quality scenarios;
    - Analyze possible conflicts among QAs and consolidate them;
    - Finally, integrate QAs into the FRs model.

The next chapter describes the application of SOQUAREM in an illustrative example.

CHAPTER 5

ILLUSTRATIVE EXAMPLE OF THE BUILDING AUTOMATION SYSTEM CASE

This chapter describes the applicability of SOQUAREM process by an example (Djouab and Suryn, 2011b). Section 1 develops the example and its operation in SOQUAREM process phases. Section 2 analyses and discusses the applicability of SOQUAREM process and finally, section 3 concludes the chapter by the resulted analysis and future improvements.

5.1 Development of the example

In this section, application of the SOQUAREM process to the MSLite system is illustrated by an example. First, the MSlite is described by its context and functional part (use case and business domain models) (Sangwan et al., 2008 and Ozkaya et al., 2008). A detailed application of SOQUAREM process to the MSLite system is then illustrated. The data describing the main inputs of SOQUAREM process, high level quality needs, BMM model and BCT table were developed. The business and refined business goals data are provided by the MSLite case. The work of Sustra (Sustra and al., 2007) was used to develop the business goal 3 (BG3) and its refined business goals. In phase 5 of the process, the data used to deal with conflict resolution among the QAs was provided (Tables 5.11 and 112).

5.1.1 Presentation of the example

The presented example has been developed from the case of Sangwan and Ozkaya (Sangwan et al., 2008) and (Ozkaya et al., 2008). These authors pinpoint the importance of quality attributes to drive the architecture of the system. They also describe how QAs are elicited from business goals. This case was selected because it provides initial data on QAs (business goals and refined goals) which help to build the example and illustrate the SOQUAREM process. Data provided from the case of Sangwan and Ozkaya include:

- Functional requirements of the MSLite system;
- The 2 business goals of MSLite system (BG1 and BG2) and their scenarios.

The remaining data in the example is provided from:

1. Djouab's research describing:
   - High level problems and quality needs;
   - BMM model and BCT concepts;
   - QAs scenarios;
   - Conflict resolution.

2. Other papers:
   - The business goal 3 (BG3) related to increasing the use of Internet (Sustar et al., 2007).

5.1.2 Description of the MSLite system

An organization wants to develop a software system called MSLite, a unified management station for a building's automation domain that will automatically monitor and/or control the internal functions of buildings, such as heating, ventilation, air conditioning, lighting, access and safety (Figures 5.1 and 5.2). The intended users of MSLite are facility managers who need to operate many (hardware) systems required to support building functions. Since there are a large number of these systems, a Field System Simulator (FSS) is used during software product development to simulate these systems.

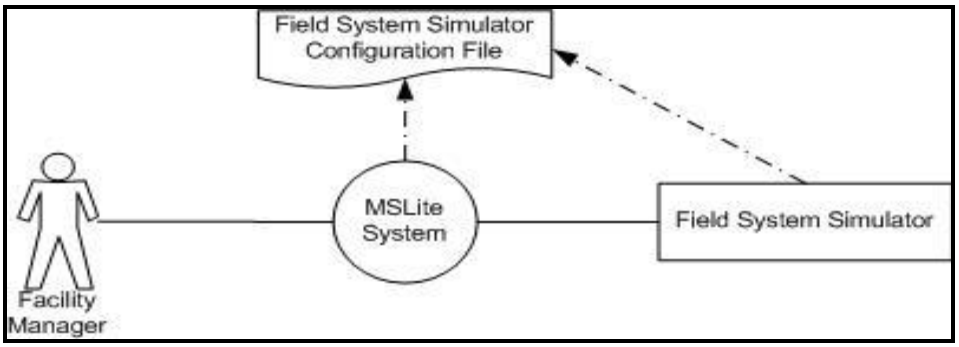


Figure 5.1 MSLite definitions
Extracted from Sangwan et al., (2008)

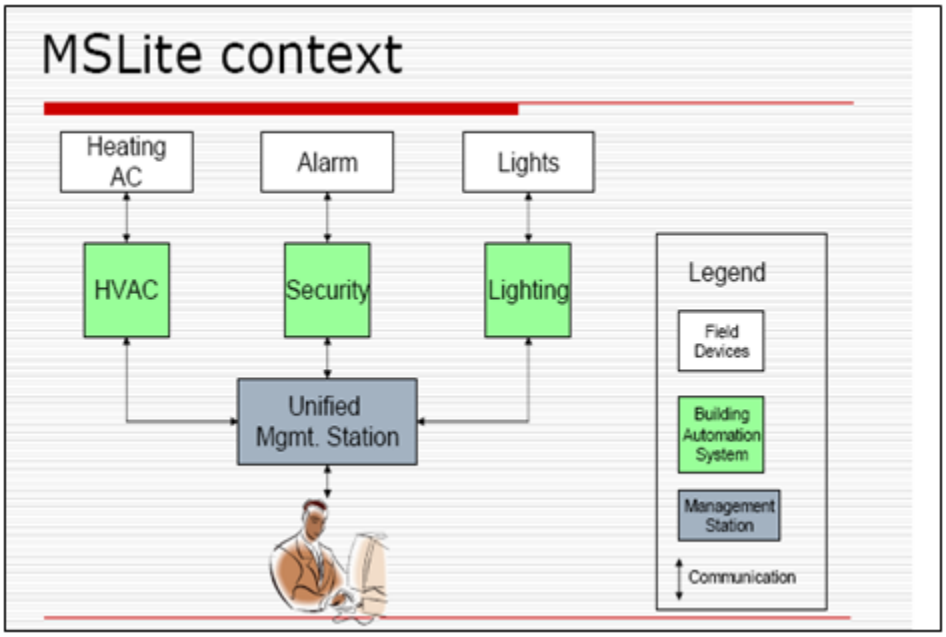


Figure 5.2 MSLite system context
Extracted from Sangwan et al., (2008)

Some of the high level functional requirements for the MSLite system are:

- **Manage** the network of **hardware-based field systems** represented in FSS used for controlling building functions;
- **Issue commands** to configure the field systems and change the values of their properties;
- **Define rules** based on property values of field systems that trigger reactions and issue commands to reset these property values;
- **Define alarm** conditions similar to rules that, when met, trigger alarms notifying the appropriate user of life-critical situations.

Figure 5.3 shows a subset of use cases and actors identified from the analysis of some of the business process to be supported by «MSLite». These use cases are listed as follows:

- Define automation rules;
- Define alarms;
- Define the SOP “Standard Operating Procedures”;
- Issue commands to field devices;
- Handle alarms and their life cycle;
- Generate alarms originating from field systems;
- Notify a change of value: including for example the changes of some field system property values and failure reports.

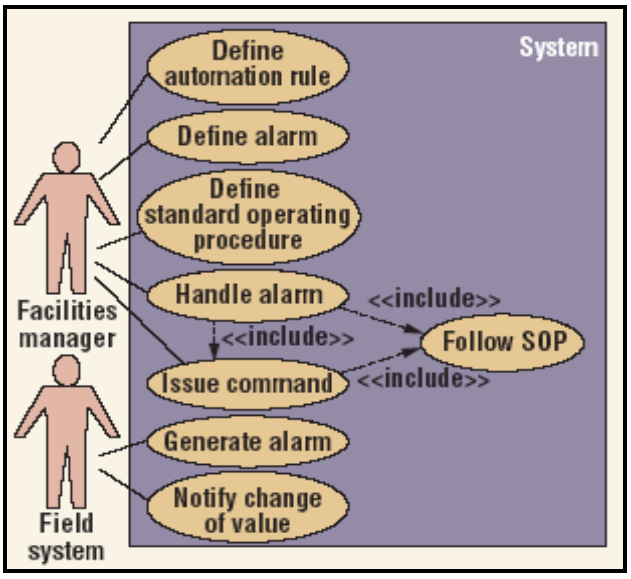


Figure 5.3 Use cases
Extracted from Ozkaya et al., (2008)

The business process descriptions in Figure 5.4 illustrates problems in the domain model of the building automation and introduce the important business entities that would be manipulated by the use cases (alarms, rules, commands and SOP (Standard Operating Procedure)).

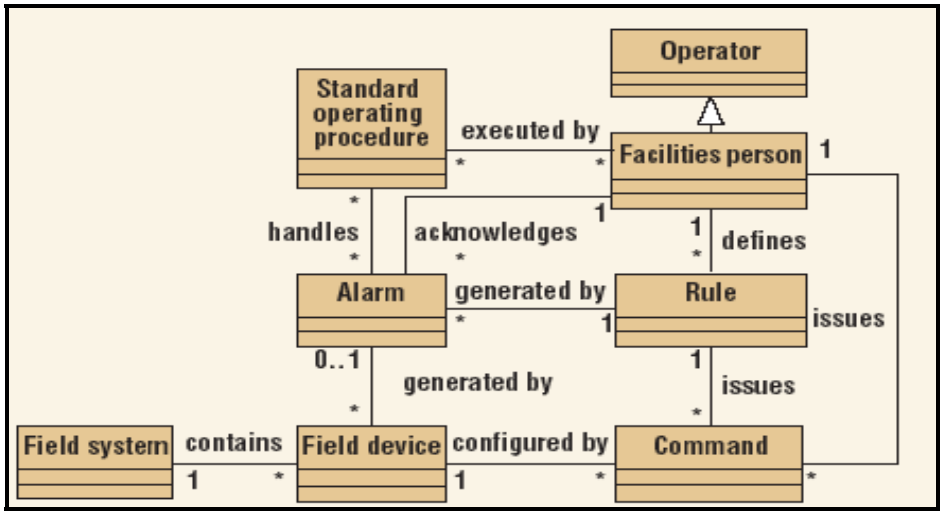


Figure 5.4 Business domain model
Extracted from Ozkaya et al., (2008)

5.1.3 Specific features of application of SOQUAREM method

In this section, application of SOQUAREM for the MSLite System is described. First, the "Why SOQUAREM process is applied to the MSLite system" is presented by summarizing high level problems and quality needs for the MSLite system. Second, the "How SOQUAREM process is applied to the MSLite system" is discussed by describing the main concepts/rules and phases of SOQUAREM process. Table 5.1 summarizes the application of SOQUAREM process to the MSLite system.

Table 5.1 SOQUAREM process applied to MSLite system

<table>
<tr><th colspan="5">SOQUAREM case study: MSLite system</th></tr>
<tr><td colspan="5">1. High level problems and quality needs for the MSLite system (Table 5.3 of BCT, section WHAT)
1.1 High level MSLite system problems: lack of web tools.
1.2 High level quality needs: operability, security.</td></tr>
<tr><td colspan="5">2. Description of SOQUAREM process for the MSLite system
2.1 Description of the main concepts of SOQUAREM: BMM (Business Motivation Model) and BCT (Business Context Elements);
2.2 Phases of SOQUAREM process.</td></tr>
<tr><td>Phase</td><td>Description</td><td>Key concepts/Rules</td><td>Input</td><td>Output</td></tr>
<tr><td>1</td><td>State business goals</td><td>BMM and BCT
Consensus session</td><td>BMM, BCT, Use case model</td><td>Business goals</td></tr>
<tr><td>…..</td><td></td><td></td><td></td><td></td></tr>
<tr><td>5</td><td>Analyze and consolidate QAs</td><td>Impact matrix
Weighted method
Consensus session</td><td>Utility tree of QAs scenarios</td><td>(QAs) template
Consolidated utility tree of QAs</td></tr>
</table>

5.1.3.1 High level problems and quality needs for the MSLite system

High level problems

Some of the high level problems for the MSLite system are summarized in the following points:

- MSLite UI is not customized to most recognized languages;
- Absence of web and communication tools;
- MSLite system does not support field systems from different manufacturers.

MSLite problems are caused by the following reasons:

- Increased use of Hardware's commoditization;
- Lack of technological platforms (as the .NET platform and the C# language).

MSLite problems have the following consequences:

- MSLite system is not efficient and not profitable;
- Shrinking profit margins.
- Unsatisfied customers;
- Loss of money.

High level quality needs

From the identified problems, some of the high level quality needs are summarized as follows:

- The overall vision for the organization is to broaden the market base by being an open general-purpose management station that can be used with a wide variety of field systems (including eventually third party (**Adaptability**).
- Build an accessible building automation system product («MSLite») with modern technologies that provides excellent user experience to satisfy advanced expectations by customers **(Usability and Adaptability and Satisfaction).**

- The system should also use web browser interfaces, which may even include building this capability into the individual controllers. The MSLite components should be designed to use internet communications for sharing information with the rest of the system. Internet-based communications should be specified to improve building operators' access to the system and to improve system communications (**Operability, Interoperability, Security and Adaptability).**
- The management station is deployed in a critical environment and must satisfy increased availability and security requirements (**Security).**
- The management station must be deployable in environments with four figure user numbers (**Adaptability**).

5.1.3.2 Description of SOQUAREM process

SOQUAREM process is applied to the MSLite system to deal with its high level technological problems and to meet associated high level quality needs. Inputs of SOQUAREM process are: Functional requirements (FRs), the BMM and BCT concepts and the main output is the list of identified quality attributes for MSLite system (Figure 5.5).

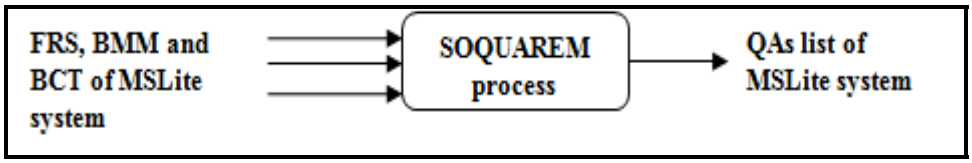


Figure 5.5 Output of SOQUAREM process applied to MSLite system

Main Key concepts to be applied in SOQUAREM phases

Figure 5.6 shows and excerpt from the developed BMM (Business Motivation Model) in the automation building system case. The desired outcome of the business transformation is represented in the frame "*Desired_Result"* which is to "offer the automation system product MSLite in new and emerging geographic markets". The "*Desired_Result"* is supported by the "*Course_ of_ Action*: opening sales channels" which is a component of the "*Mission:* reduce total development costs for the management stations and coordinate sales channels"

that make operative the "*Vision:* broaden market base with an open general-purpose management station that can be used with a wide variety of field systems".
From the "*Desired_Result"* frame, emerge two major business goals:

1. **Goal 1**: Be a market leader by supporting the system with additional language features, cultures and regulations;
2. **Goal 2**: Use a third part seller, the "Value Added Resellers" to increase sales.

The two goals are supported by four strategies (Table 5.2).

Directives to support achievement of the desired results are both federal, provincial and building system specific directives. Directives are divided into three categories: act/legislation, policy and agreement (Table 5.2).

Influencers are an important item in the BMM. They have a strategic influence on the building automation system. They are represented in Figure 5.5 with the building automation system's **assessments** of them. ***Influencers*** could present strengths and weaknesses, opportunities and threats (Table 5.2). Table 5.3 (Business Context Table BCT) describes business context elements for the MSLite system.

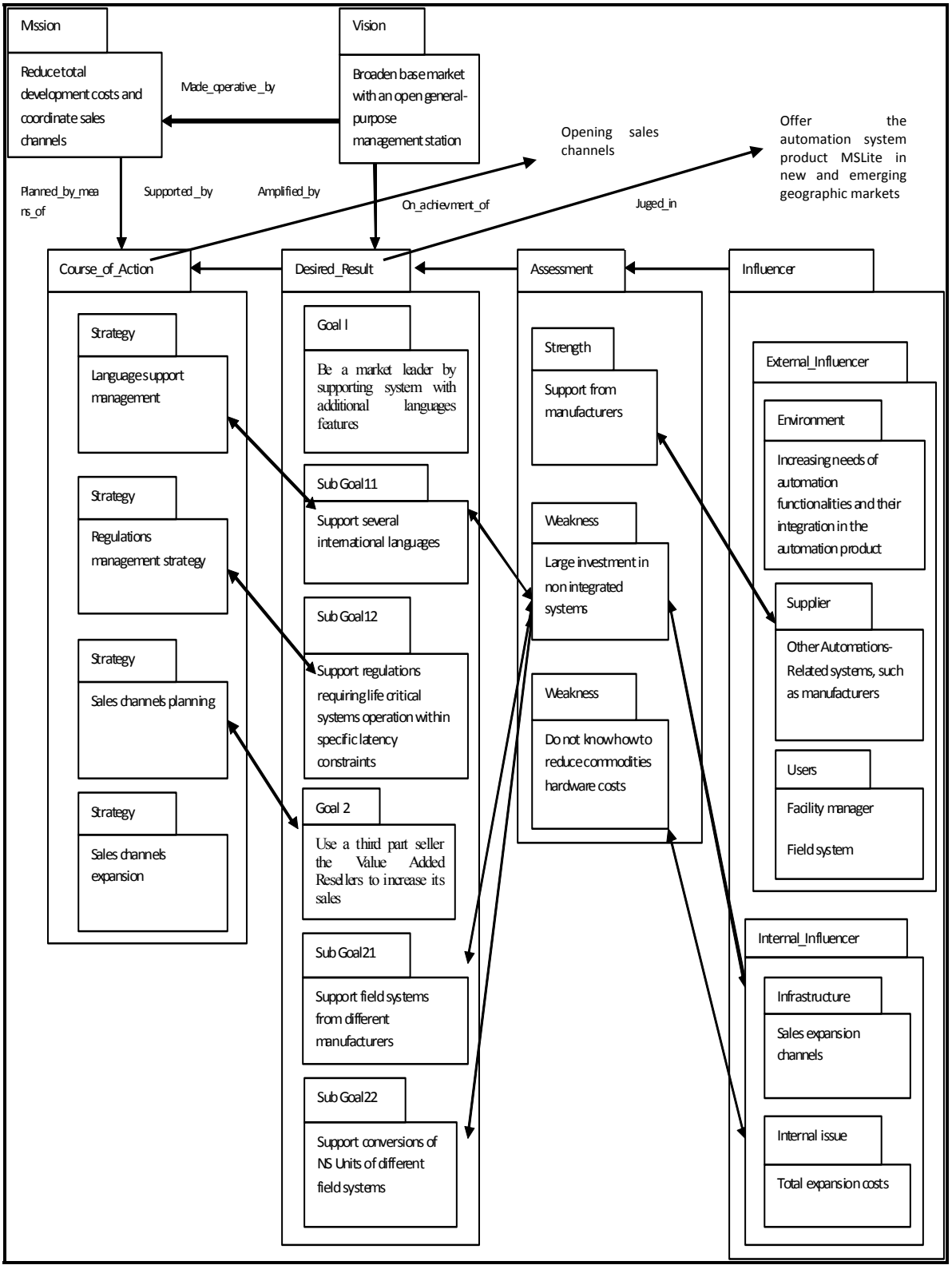


Figure 5.6 BMM for automation building system

Table 5.2 BMM concepts used for automation building system

BMM concepts	Description
Vision	***Desired Result*** 1. Goal1: Be a market leader by supporting system with additional language features, cultures and regulations; 2. Goal2: Use a third part seller the "Value Added Resellers" to increase its sales.
Mission	***Course of Action*** 1. The "sales channels planning" strategy efforts towards achieving the second goal. 2. The "sales channels expansion strategy" is aiming to allow sales channels to be done through the "Value Added Resellers" in diverse locations and manufacturers types. 3. The two strategies "language support management for the MSLite" and "Regulations management" are developed for the first goal to allow support international languages and different regulations into the MSLite system.
Directives	1. The insurance act of the "Building automation system" oversees the whole building automation system processing and is linked to goals "Goal1" and "Goal2"; 2. The privacy protection act protects usage of building automation system information. Its strategy is to regulate and guide the management of building automation system and information; 3. The interprovincial sales channels agreements are helpful in achieving the first goal "Goal1" of entering new and emerging geographic markets.
Influencers	1. External influencers include: Suppliers and manufacturers. Facilities managers, field system and resellers are main actors in the building automation process; 2. Internal influencer could be Legacy information System, sales expansions channels and total expansion costs.
Assessments	1. Strength: a. Support from manufacturers 2. Weaknesses: a. Large investment in non integrated systems b. Unknown hardware commodities cost.

Table 5.3 BCT for automation building system

	Business context elements
What	1. **Business goals** a. Enter new emerging geographic markets; b. Expand sales channels through value added resellers. 2. **High level problems and technological constraints** a. MSLite UI is not customized with most recognized languages; b. Absence of web and communication tools; c. MSLite system do not support field systems from different manufacturers; d. Implementation language will be 'C#' and the implementation platform will be '.NET'; e. System will support a management station software to manage the field systems; f. System will be modified according to market's languages, cultures and regulations; g. Application will feature a HTML based web user interface and compatible at least with Internet Explorer 5.5; h. The integration of commercial off-the-shelf components is not possible due to budget considerations. 3. **High level quality needs** a. The overall vision for organization is to broaden market base by being an open general-purpose management station that can be used with a wide variety of field systems (including eventually third party (**Adaptability**); b. Build an accessible building automation system product MSLite with modern technologies that provides excellent user experience to satisfy advanced expectations by customers **(Usability and Adaptability and Satisfaction);** c. The system should also use web browser interfaces, which may even include building this capability into the individual controllers. The MSLite components should be designed to use Internet communications for sharing information with the rest of the system. Internet-based communications should be specified to improve building operators' access to the system and to improve system communications (**Operability, Interoperability, Adaptability and Security);** d. The management station is deployed in critical environment and must satisfy increased availability and security requirements (**Security);** e. The management station must be deployable in environments with four figure user numbers (**Adaptability**). 4. **High level functional requirements** a. Manage the network of hardware-based field systems represented in FSS used for controlling building functions; b. Issue commands to configure the field systems and change values of their properties; c. Define rules based on property values of field systems that trigger reactions and issue commands to reset these property values;

Table 5.3 BCT for automation building system (follow)

	Business context elements
What	d. Define alarm conditions similar to rules that when met trigger alarms notifying appropriate user of life-critical situations. 5. **Regulations and compliance** a. Certain regulations require all life critical systems to operate within specific latency constraints. The system must be able to meet these latency requirements with a sufficient margin. 6. **Domain characteristics** a. Context of the system**:** Unified management system for the Building automation system of different field devices; b. Field devices**:** alarms, heating, ventilation, air conditioning, lighting, access and safety. 7. **Political interests and organizational culture** a. Political interests: oriented towards a more recognized and unified management system; b. Organizational culture: putting emphasis on flexible employers.
How	**Business strategies to achieve business goals** a. Modern technologies based on useful GUI that satisfy advanced expectations of customers; b. Channels planning Strategy; c. Sales channels expansion strategy.
Who	**Target stakeholders** a. Facilities manager; b. Field system; c. Resellers and building automation system user.
Why	1. **Current business** a. Outcome: Improve profit margins and be market leader in automations systems 2. **Needs for target stakeholders to be met** a. Control and monitor building functionalities in a way that ensures functionality, efficiency, privacy, reliability and simplicity. 3. **Business mandate** a. Enter new emerging geographic market by modifying system to support different languages, cultures and regulations. Languages could be non Latin characters and scripts written from right to left and supporting regulations that require life critical systems to operate within specific latency constraints; b. Expand sales channels through value added resellers and support hardware devices from different manufacturers. Support also conversions of non standard units used by different field systems for rule evaluation and commands without errors and user intervention.

Description of SOQUAREM phases

Phase 1: State and identify the business goals of the system (Table 5.4)

- An organization wants to extend its automation system product MSLite in:
  - New and emerging geographic markets;
  - Expand sales channels through value-added resellers by letting resellers sell the software system under their own brands.
- Resellers would support field systems from the manufacturers they choose.
- Figure 5.7 shows the concepts involved in this phase:
  - BMM to define business goals by its "desired results" frame;
  - BCT (Why and What questions) to structure and organize business goals;
  - Consensus session to confirm business goals with stakeholders;
  - Statement rules to verify if business goals are correctly defined.
- By applying the statement rule STR1(Table 4.10): "*Each business goal is defined according to the* "**Business mandate"** item of **BCT (**Table 5.2) *item* and *the* "**Desired results"** item of **BMM** (Table 5.3), 2 business goals **BG1** and **BG2** are defined (Figure 5.8).

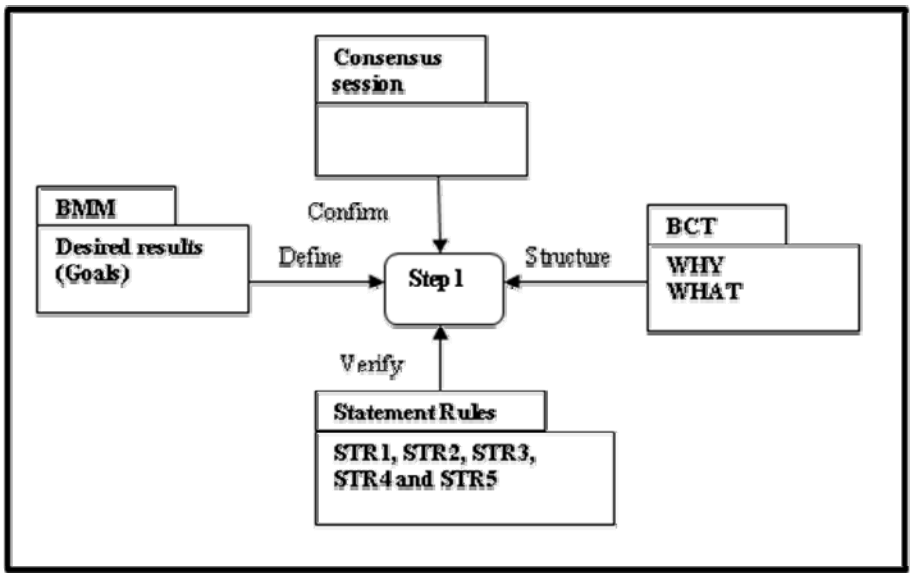


Figure 5.7 Concepts of phase 1

Table 5.4 State and identify business goals for MSLite system

Step1: State and identify business goals of the system		
Input	**Statement rules**	**Output**
BCT (WHY) (Table 5.3) 1. Business mandate: 2. Target stakeholder's needs **BCT (WHAT)** 1. Domain characteristics 2. High level problems and technological constraints 3. High level quality needs 4. High level FRs 5. Political interests and organizational culture **BMM (Desired results): (Table 5.2)** 1. Desired results (Goals): a. Be a market leader by supporting system with additional languages features, cultures and regulations. b. Use a third part seller the Value Added Resellers to increase its sales	1. STR1 2. STR2 3. STR3 4. STR4 5. STR5	**Business goals (BGi)** 1. BG1: Enter new emerging geographic markets 2. BG2: Expand sales channels through value added resellers

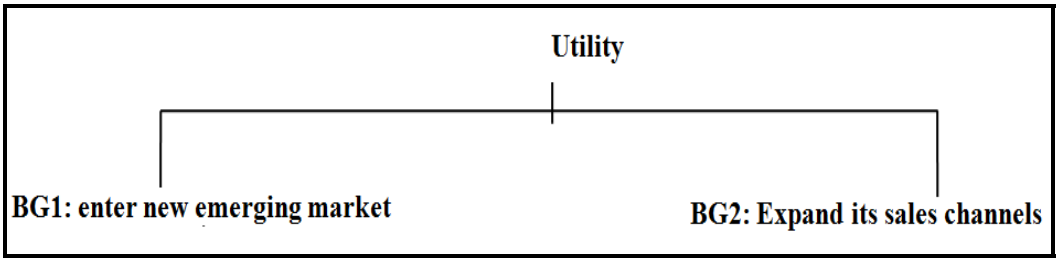


Figure 5.8 Business goals of the MSLite system

Phase 2: Refine the business goals (Table 5.5)

- MSLite was developed to support a wide variety of field systems (including an eventual third party), international languages and regulations constraints.

- MSLite should use web browser interfaces and communication tools for sharing information with other computer applications such as online weather-forecasting services to improve building operators' access to the system.
- Use of an internet communications protocol XML may allow MSLite system to seamlessly communicate with business enterprise software such as accounting and business scheduling packages (Figure 5.10).
- Business goals are detailed according to additional business information such as organizational culture, regulations and guidelines, technological constraints and business strategies. Figure 5.9 shows the concepts involved in this phase:

    - BMM defines refined business goals by its “course of action” and “directives” frames;
    - BCT (How and what questions) to structure and organize refined business goals;
    - Consensus session to confirm refined business goals with stakeholders;
    - Refinement rules to verify if refined business goals are correctly detailed.

- By applying the refinement rule RFR1 (Table 4.11): “Each business goal is detailed according to **technological constraints**, existing **regulations and compliance** and high level **functional requirements**”. The refined business goals are described with their priority in the “Refined business table**”** (Table 5.6). Business goals and their refined goals are represented in the utility tree (Figure 5.11).

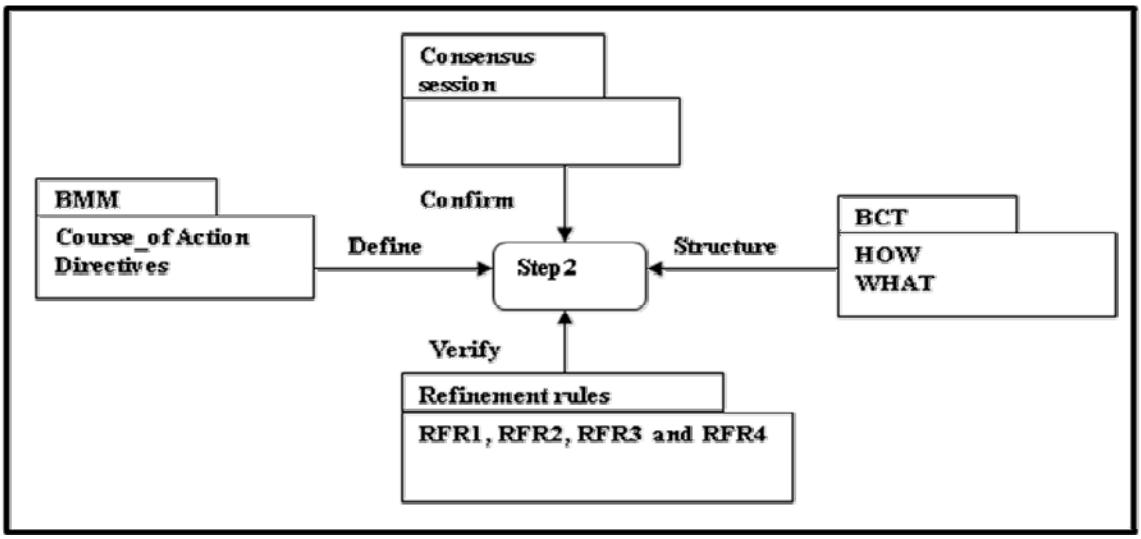


Figure 5.9 Concepts of phase 2

Table 5.5 Refine business goals

Step2: Refine business goals of the system		
Input	**Refinement rules**	**Output**
Business goals: BG1, BG2 and BG3 **BCT (HOW) (Table 5.3)** a. Business strategies to achieve business goals **BCT (WHAT):** a. Technological constraints b. High level functional requirements c. Regulations and compliance **BMM (Table 5.2)** a. Courses of actions (strategies) b. Directives	1.RFR1 2.RFR2 3.RFR3 4.RFR4	**Refined business goals (Table 5.6)** 1. BG1: Enter a new emerging geographic market a. BG1.1: Support several international languages i.BG1.1.1. b. BG1.2: Support regulations to operate within specific latency constraints 2. BG2: Expand its sales channels through value-added resellers a. BG2.1: Support field systems from different manufacturers b. BG2.2 i.BG2.2.1. 3. BG3: Increase use of Internet a. BG31: Support the emerging standard XML for «MSLite» i.BG3.1.1.

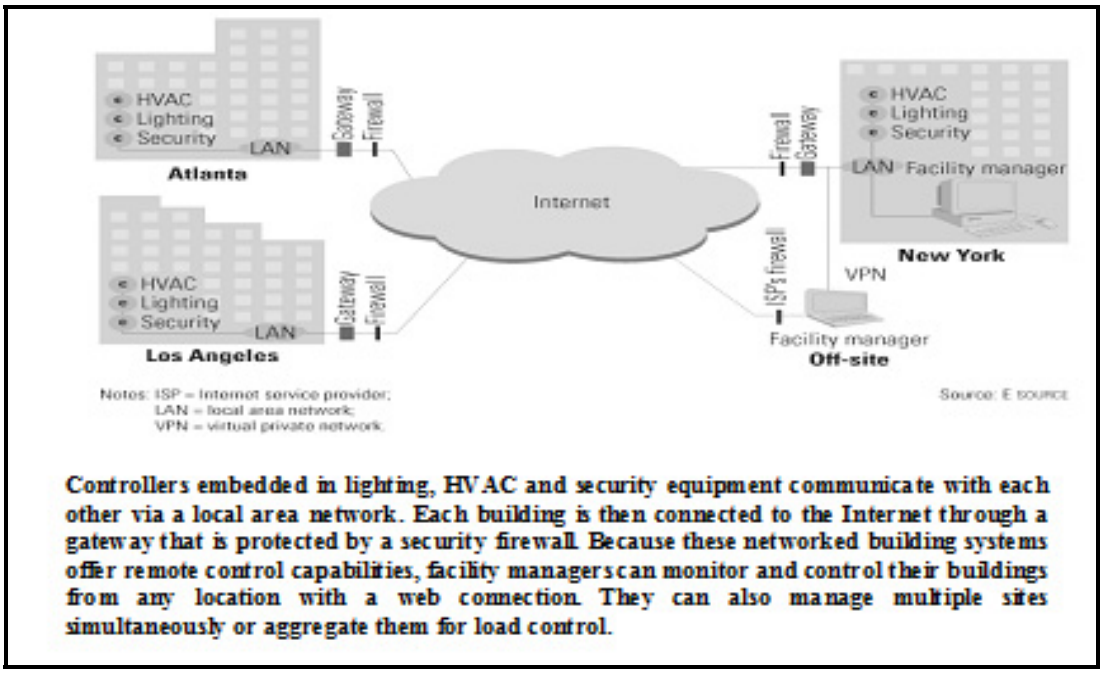


Figure 5.10 How a web browser interface works
Extracted from Sustar et al., (2007)

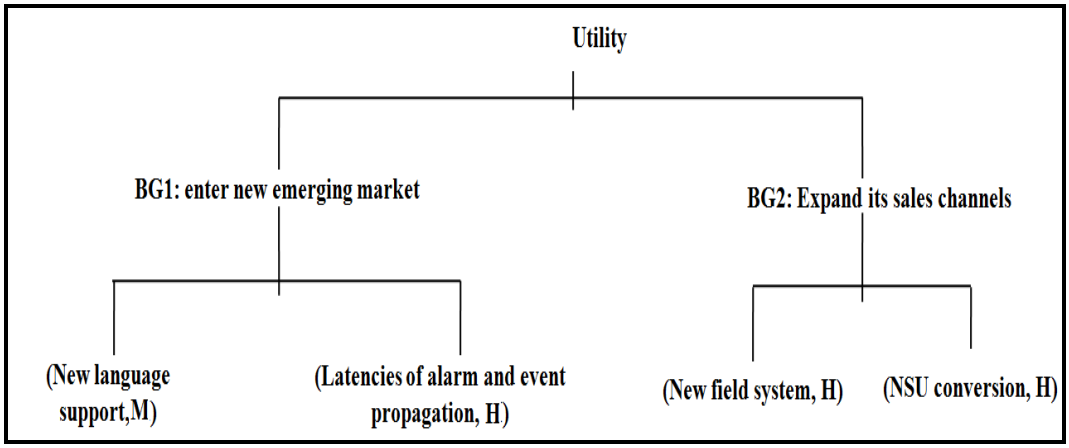


Figure 5.11 Refined business goals of the MSLite system

Table 5.6 Refined business goals table

Business goal	Goal refinement	Priority
BG1: Enter a new emerging geographic market:	It must be possible to modify the system to support different languages, cultures and regulations. **BG1.1:** Support several international languages **BG1.1.1:** The system must allow changing all user interactions language to a language of choice. This includes languages with non-Latin characters and scripts written from right to left. **BG1.2:** Support regulations that require life-critical systems, such as fire alarms, to operate within specific latency constraints **BG1.2.1:** Certain regulations and certifications require all life critical systems such as fire alarms and intrusion detection systems to operate within specific latency constraints. The system must be able to meet these latency requirements with a sufficient margin.	 **M** **H**
BG2: Expand its sales channels through value-added resellers.	To succeed in the Value Added Resellers market, the system must be able to support hardware from different manufacturers. This includes existing and to some extent future devices. **BG2.1:** Support field systems from different manufacturers **BG2.2:** Support conversions of nonstandard units used by the different field systems **BG2.2.1**: The field devices supported by the system can use different units. These units can be different from the units used by the user when specifying automation rules thresholds and commands. The system must be able to make all required conversions for rule evaluation and commands without errors and without user intervention	 **H** **H**
BG3: Increase use of Internet: Use web browser interface which usually runs on a dedicated web server.	Web browser interface allows a user to access and view the MSLite through the Internet using a computer that is running web browser software. Users can take advantage of this capability to monitor and control the MSLite in multiple facilities from a single computer **BG31:** Supporting the emerging standard XML for «MSLite», manufacturers give their customers the flexibility to configure the system on their own, use a configuration package from another manufacturer, or use a third-party software package that supports XML as a file format, such as Microsoft Excel and Microsoft Access. Because Microsoft is freely distributing its XML software engine, it's much easier for manufacturers, software developers, or users to create custom applications that read and write XML data, possibly even reading proprietary configuration data files and exporting them in standard XML format. **BG311:** The BACnet standard of MSLite will be added with XML and web services in order to exchange data with other computing applications over a network. One initial use of web services is to enable sophisticated functionality, such as creating "virtual thermostats" that give users control over the temperatures in their own areas. Use also web services to integrate BASs with utility systems, which would implement control strategies based on real-time pricing.	H

Phase 3: Link the business goals to corresponding quality attributes (Table 5.7)

Identify quality attributes of the MSLite system (**Adaptability** and **Efficiency**) and relevant actors and actions to achieve them (QAs list in Table 5.8). Figure 5.12 illustrates the concepts involved in this phase:

- BCT concept:
  - WHO: **target stakeholders** to define **relevant actors** related to the QA;
  - WHAT: **high level quality needs** to define the candidate quality attributes of the system.
- BMM concept:
  - **External Influencer** to define **relevant actors** related to the QA;
  - **Internal Influencer** to identify **relevant actions** to achieve the QA.

- ISO/IEC 25030 used to infer the right quality attribute;
- Refined business goals to help identify relevant actions of the QA;
- Linkage rules to verify if QAs are correctly identified.

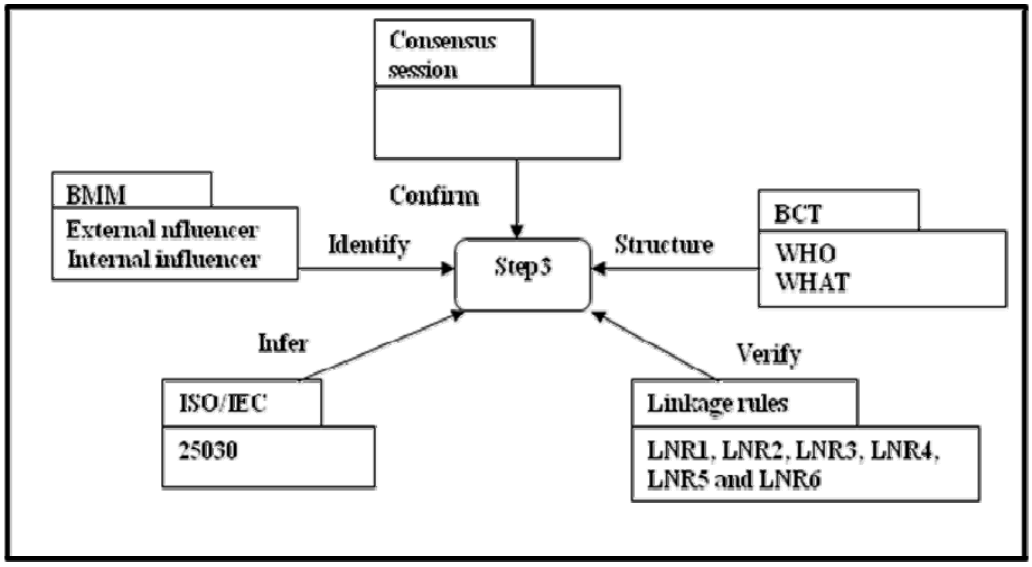


Figure 5.12 Concepts of phase 3

Table 5.7 Link the business goals to the corresponding quality attributes

Step3: link the business goals to the corresponding quality attributes		
Input	**Linkage rules**	**Output**
Refined business goals; **BCT (WHO) (Table 5.3)** a. Target stakeholders **BCT (WHAT) (Table 5.3)** b. High level quality needs c. ISO/IEC 25030 standard **BMM** a. External influencer b. Internal influencer	1. LNR1 2. LNR2 3. LNR3 4. LNR4 5. LNR5 6. LNR6	**Quality attributes list (Table 5.8)** a. Adaptability b. Efficiency **Relevant actors (Table 5.8)** a. Field system b. Facility manager **Relevant actions (Table 5.8)** a. Add new language b. Modify language c. Report alarm d. Update change to property value e. Notify change of property value f. Add new field system g. Handle non standard units

By applying the linkage rule LNR1 (Table 4.12): "*Each QA is derived according to* ***high level quality needs****, the* ***refined business goals,*** *the* ***target stakeholders*** *and* ***ISO/IEC 25030***", two QAs have been identified (Figures 5.13 and 5.14) (**Adaptability** and **Efficiency**) and their **actors** responsible to achieve them (**facility manager** and **field system**).

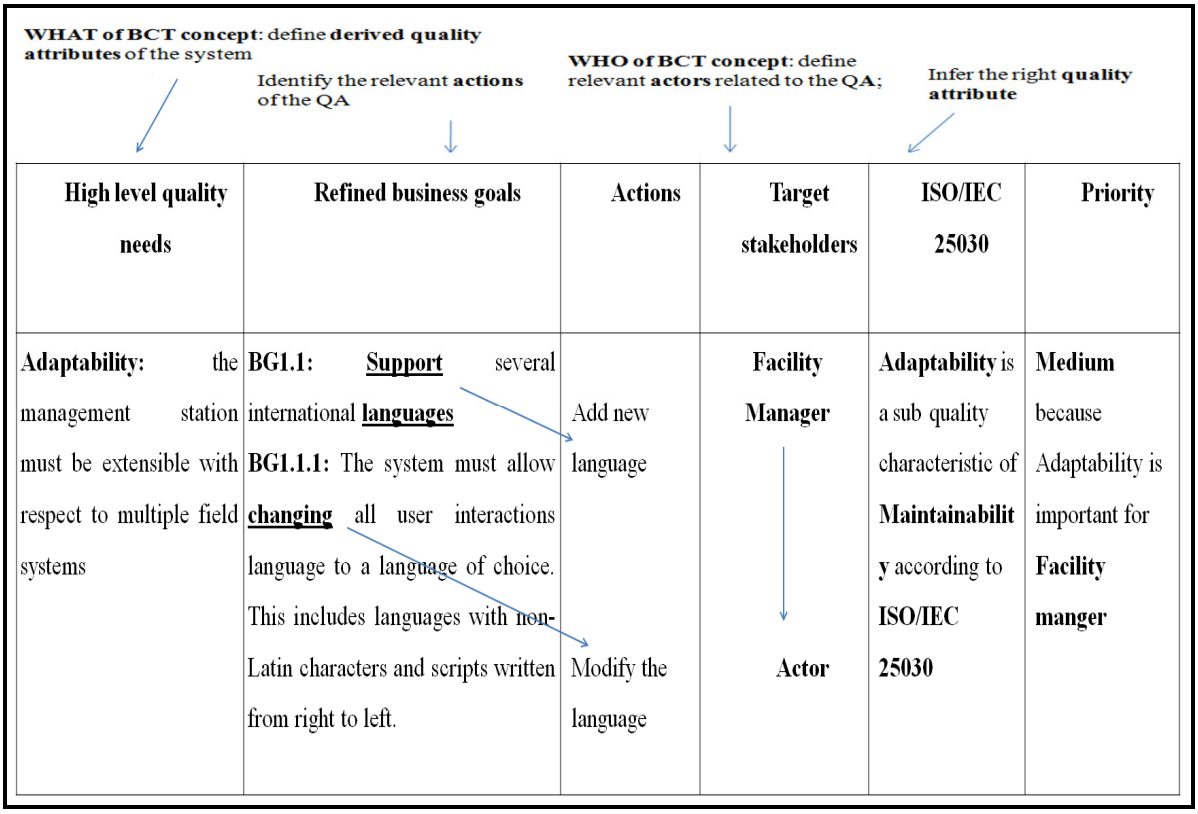


Figure 5.13 Application of the first linkage rule LNR1

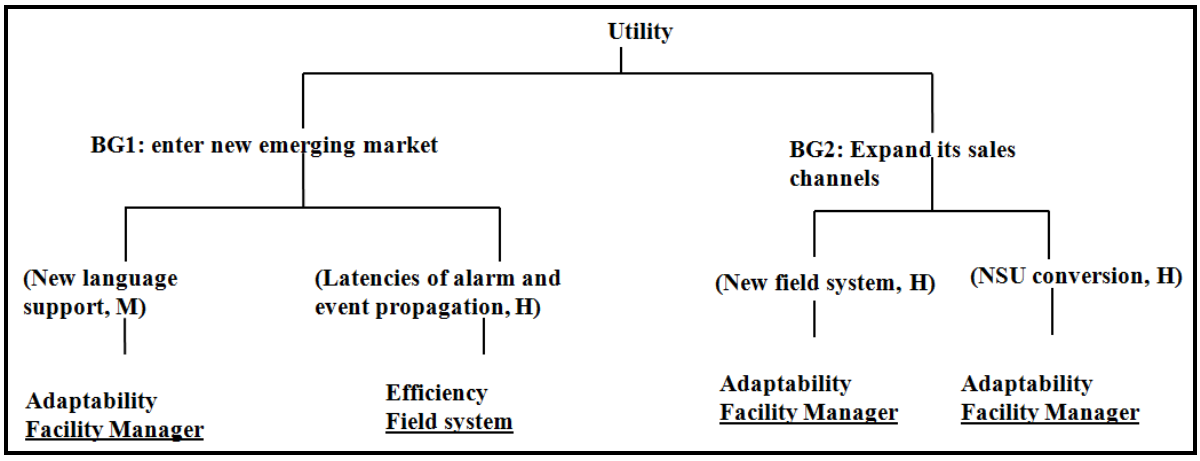


Figure 5.14 QAs and their respective actors

Table 5.8 Quality attributes list

Business goals	Refined business goals	Priority	Derived QA	Relevant actors	Actions
BG1: Enter a new emerging geographic market.	**International language** BG11: Support several international languages BG1.1.1: The system must allow changing all user interactions language to a language of choice. This includes languages with non-Latin characters and scripts written from right to left.	M	Adaptability	Facility manager	Add new language Modify the language
	Latencies of alarm and event propagation BG12: Support regulations that require life-critical systems, such as fire alarms, to operate within specific latency constraints	H	Efficiency	Field system	Update a change in property value in all UI screens Notify property value to the MSLite system Report the life-critical alarm to the concerned users within 3 seconds of the occurrence of the event that generated the alarm
BG2: Expand its sales channels through value-added resellers	**New field device system** BG2.1: Support field systems from different manufacturers	H	Adaptability	Facility manager	Add new field system
	Non-standard units BG2.2: Support conversions of nonstandard units used by the different field systems	H	Adaptability	Facility manager	Handle unit from the added field device

Table 5.9 illustrates the confirmed quality attributes linked to business goals with interested stakeholders (developer and business manager) during the application of the consensus session. C/R is an abbreviation of Confirmed/Rejected.

Table 5.9 Confirm linkage of quality attributes with business goals

Consensus session Description: confirm linkage of quality attributes with business goals Stakeholders involved: Manager, developer					
Business goals	**BG1:** Enter a new emerging geographic market.		**BG2:** Expand its sales channels through value-added resellers		**BG3:** Increase use of the Internet
Refined business goals	**BG11:** Support several international languages	**BG12:** Support regulations that require life-critical systems to operate within specific latency constraints	**BG21:** Support field systems from different manufacturers	**BG22:** Support conversions of nonstandard units used by the different field systems	**BG31:** Support the emerging standard XML for MSLite and provide customers flexibility to configure the system on their own applications.
Quality attributes	Adaptability	Efficiency	Adaptability	Adaptability	Operability and Adaptability
	C	C	C	C	C

Phase 4: Build the QAs scenarios (Figure 5.15 and Table 5.10)

Build quality scenarios associated to the derived quality attributes by using:

- Structure of QAs scenario template (Table 4.9);
- QAs list and relevant actors and actions to achieve QAs (Table 5.8);
- The QAs scenarios are built as follows:
  - Mapping the “Action” item of the scenario template to the relevant actions of the QAs list;

- The "Asset" item of the scenario template is specified from the definition of the refined business goals (Table 5.6).

- The priority of QA is defined according to its importance for the actor;
- For the derived QA "Adaptability" associated with **BG1**, 2 scenarios are built (Figure 5.16):

  - Scenario1: (Add new language, UI);
  - Scenario2: (Modify new language, UI);

- A consensus session is applied to confirm resulted quality scenarios with stakeholders.
- A utility tree of quality attributes corresponding to the MSLite system (**adaptability and efficiency**) is represented in Figure5.17. Mapping between identified elements of BMM (desired result, external and internal influencer) and those of the utility tree (quality attributes, actors and actions) is also represented in the utility tree.

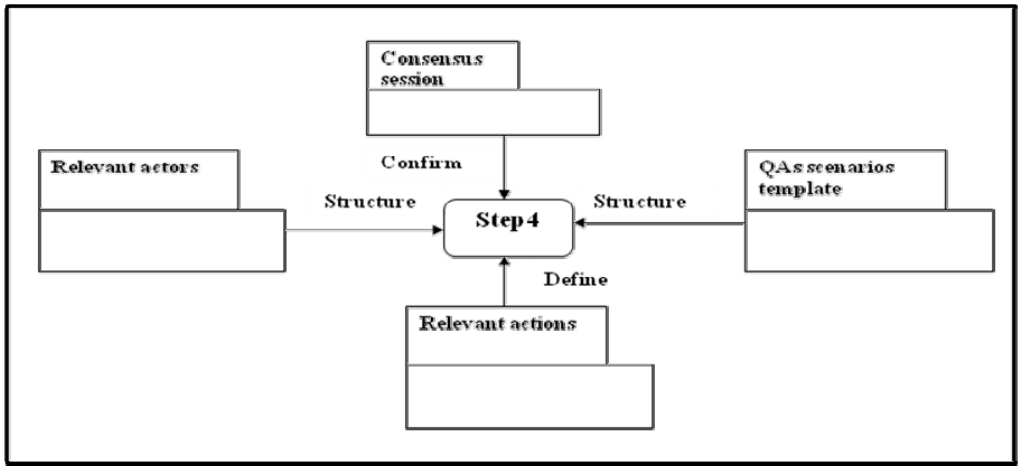


Figure 5.15 Concepts of phase 4

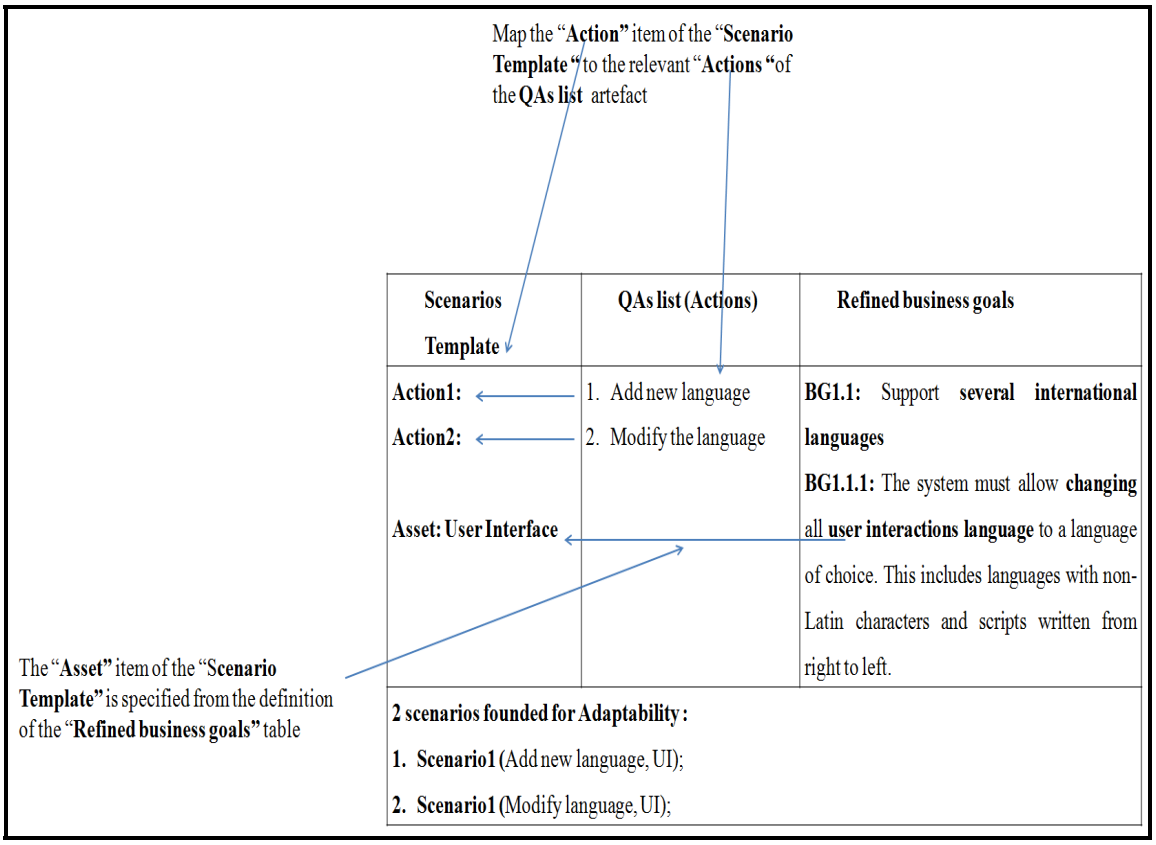


Figure 5.16 Scenarios build for Adaptability and BG1.1

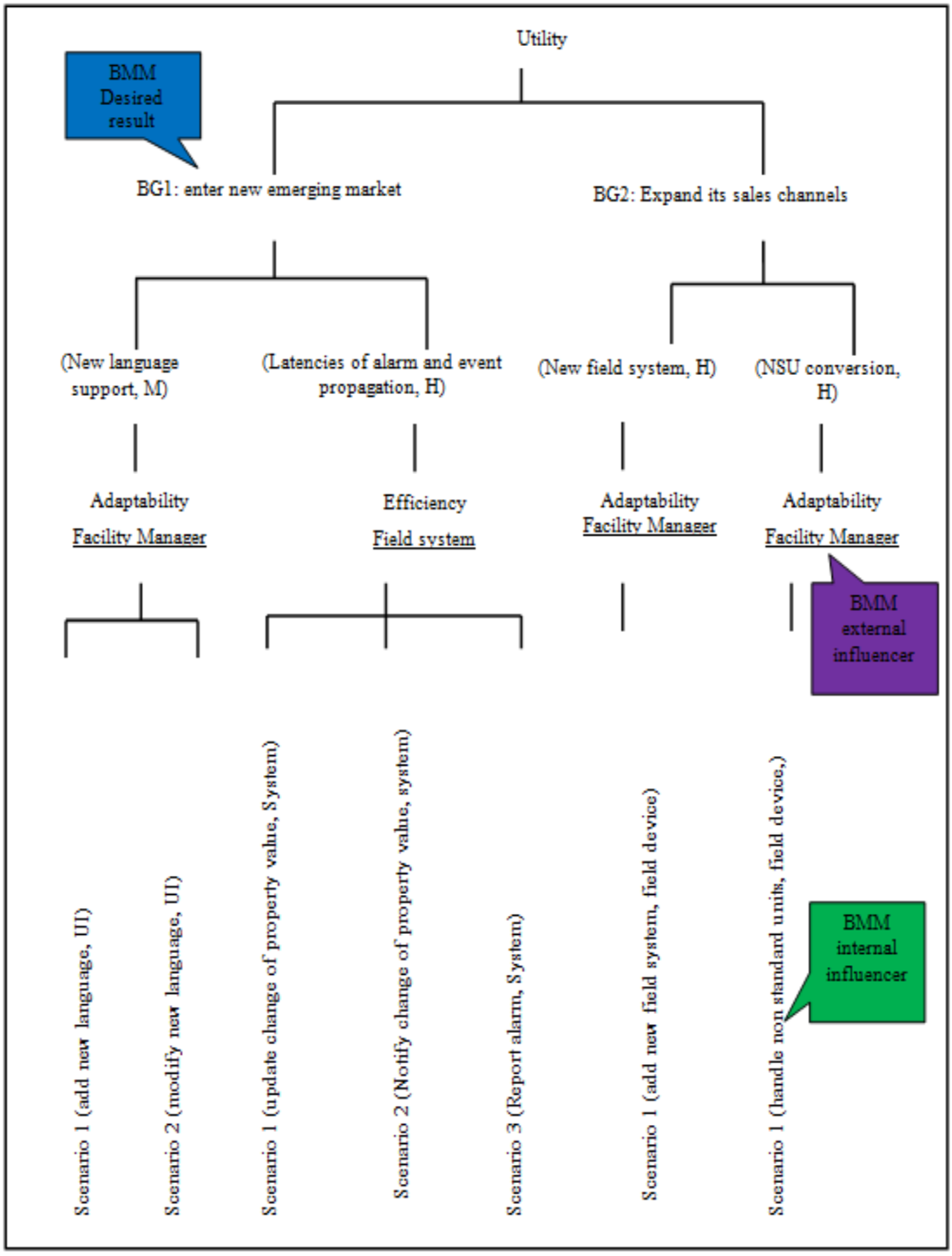


Figure 5.17 Utility tree of quality attributes

Table 5.10 Quality attributes scenarios

Derived QA, (Number of scenarios)	**Quality scenarios**
Adaptability (2)	a. The system should support new language without any required code modification. **1. Scenario 1 (add new language, UI)** **2. Scenario 2 (modify new language, UI)**
Efficiency (3)	a. A change in property value is detected by the field device and notified to the MSLite system. The value is updated in all UI screens that display the property value **1. Scenario 1 (update change of property value, system)** **2. Scenario 2 (Notify change of property value, system)** b. An event which should trigger an alarm is generated in a field device. A life-critical alarm should be reported to the concerned users and displayed on the UI of all users that must receive it. **3. Scenario 3 (Report alarm, system)**
Adaptability (1)	a. Add new field device system which should offer functionality similar to the FFS. Extend the UI of the MSLite with the new device configuration information. 1. **Scenario 1 (add new field system, field device)**
Adaptability (1)	a. Support conversion of the new connected field device (to the system) using non-SI units 1. **Scenario 1 (handle non standard units, field device)**

Phase 5: analyze conflicts among quality attributes and consolidate them (Figure 5.18)

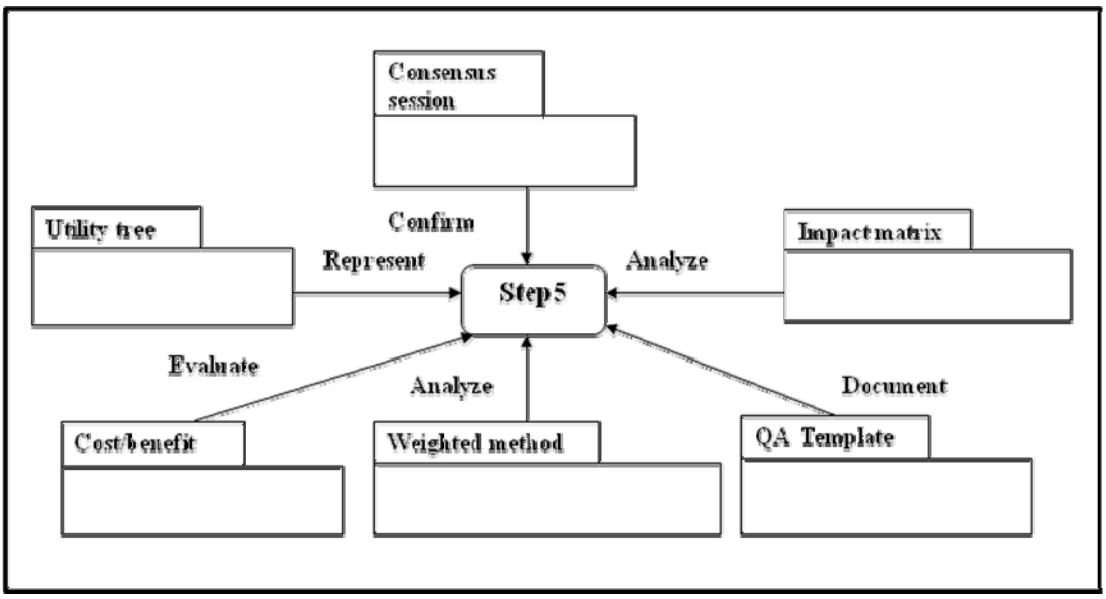


Figure 5.18 Concepts of phase 5

Some of the QAs may be found to conflict with each other while others appear to complement or strengthen one another. In this phase, interactions among quality attributes are evaluated in order to adjust the utility tree. The following steps are applied:

1. **Build Impact matrix**: shows in which way (negatively or positively) a quality attribute impacts on the others. Whenever there is a negative contribution between quality attributes there is a conflict. In this case, the efficiency of the system impacts negatively on adaptability, operability and interoperability of the system with other internet applications. Interoperability and operability may conflict with the security of exchanged data. But interoperability and operability are likely to complement the adaptability of the system to the new specifications (new language, new field system, non standard units conversion and new configuration). On the utility tree, dotted lines marked by a plus or minus signs are used to represent positive and negative interactions. The model can be used by developers to identify the most beneficial QAs with the least conflict. An example is shown in Figure 5.19.

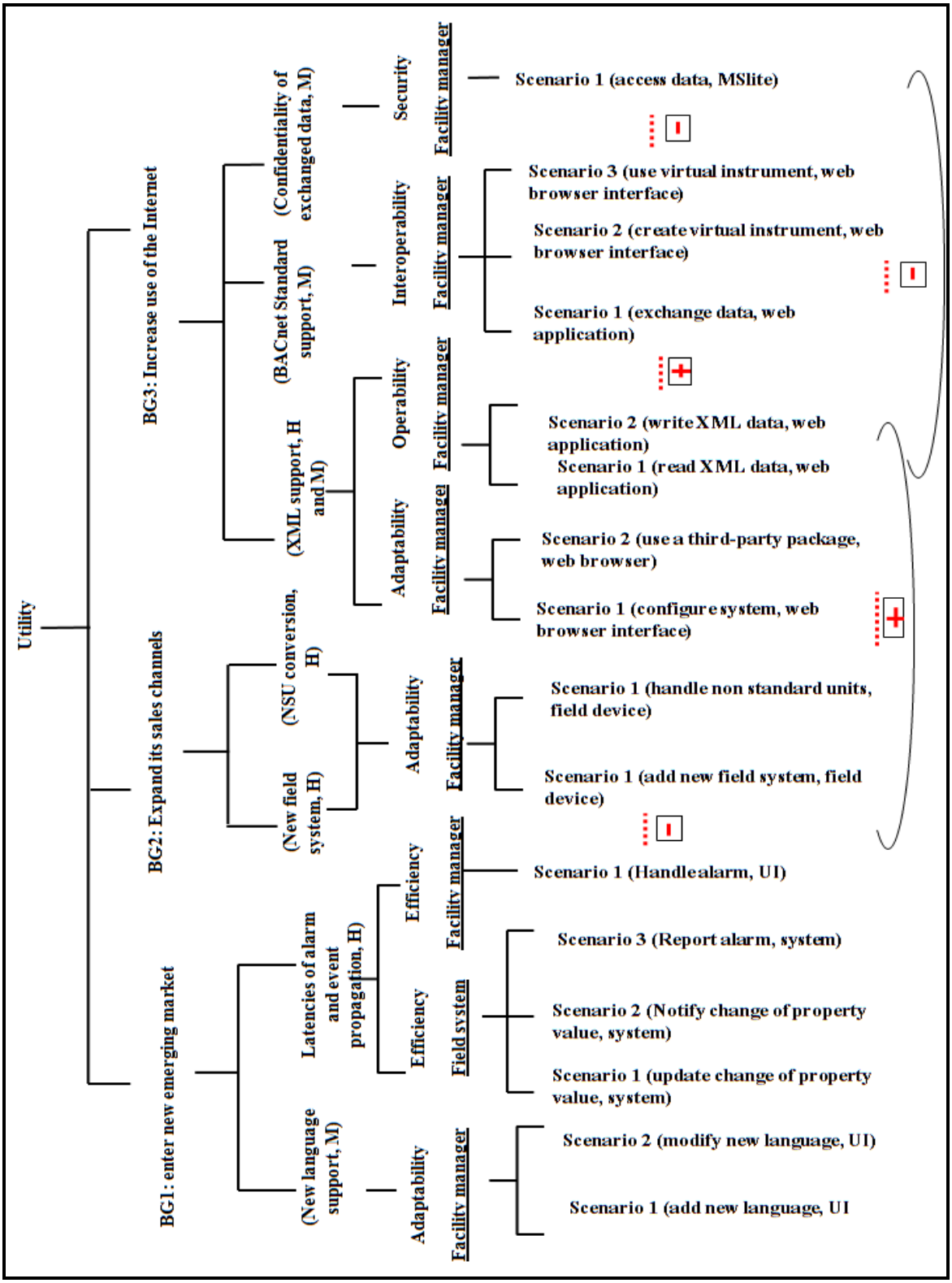


Figure 5.19 Utility tree with conflicts

2. **Attribute weights to conflicting quality attributes (Table 5.11)**: conflicts among quality attributes could be resolved by attributing weights to the cells of the quality attribute/actor matrix where the conflicting quality attributes apply to the same actors. The values are given by the "developer" and the "manager" according to the importance each quality attribute has for each actor. Used scales are based on the fuzzy logic [section 4.3.1, phase 5]. Using fuzzy values (very important, important, medium and low) facilitates the stakeholders' task of attributing priorities to conflicting QAs. Therefore, for an actor facility manager, for example, efficiency has a higher priority than adaptability and interoperability (except for BG21) and adaptability has higher priority than interoperability (except for BG11).

Table 5.11 Weighted method

Actor / QA/ RGBi	Facility manager	Field system
Efficiency/BG12	1,0	1,0
Adaptability	BG11: 0,6 / BG21: 1,0 / BG22: 0,8 / BG31: 0,8	
Security/BG33	0,7	
Interoperability/BG 32	0,7	
Operability/BG31	0,7	

3. Resolve conflicts

Facility manager actor shows 3 conflicting situations between:

- *Efficiency* and *adaptability;*
- *Interoperability and Security;*
- *Operability and Security.*

These kinds of quality attributes impact negatively on each other and have the same weight allocated to them (see the highlighted cells in Table 5.11). *For the first very important category of quality attributes (weighted at **1,0**),* the *facility manager* needs to handle the alarm in time and define the new field system. For the second important category of *quality attributes (weighted at **0,7**),* on one hand, the *facility manager* needs to interoperate with other internet applications and access the XML data. The third category, the *facility manager* is required to read/write XML data in a secure way. To resolve these kinds of conflict negotiation is needed among the stakeholders. One suitable solution is (Table 5.12):

- To lower the weight allocated to adaptability to **0.8** for the affected actor. This is because efficiency is more important than adaptability. It is essential that the alarm is handled at time even though the user may not see if the new field system has been added.
- To lower the weight allocated to interoperability to **0.6** because in this case it is essential to ensure the security of exchanged data before interoperating with other applications.
- To lower the weight allocated to operability to **0.5** because in this case it is essential to communicate with other applications in a secure way then read or write the XML data.

Table 5.12 Resolve conflicts among QAs

<table>
<tr><th>Actor
QA/ RGBi</th><th>Facility manager</th><th>Field system</th></tr>
<tr><td>Efficiency/BG12</td><td>1,0</td><td>1,0</td></tr>
<tr><td>Adaptability</td><td><table><tr><th>BG11</th><th>BG21</th><th>BG22</th><th>BG31</th></tr><tr><td>0,5</td><td>0,8</td><td>0,8</td><td>0,8</td></tr></table></td><td></td></tr>
<tr><td>Security/BG33</td><td>0,7</td><td></td></tr>
<tr><td>Interoperability/BG 32</td><td>0,6</td><td></td></tr>
<tr><td>Operability/BG31</td><td>0,5</td><td></td></tr>
</table>

In summary, *operability* is judged to have less priority than security and i*nteroperability*, so it may be acceptable to have operability partially satisfied in order to achieve satisfaction of the security and *interoperability* NFRs, as shown in Figure 5.20 (labels (P) for partially satisfied and (S) for satisfied). The final utility tree to be linked with the functional process is presented in Figure 5.21. Table 5.13 shows the QAs description template for the e*fficiency* QA.

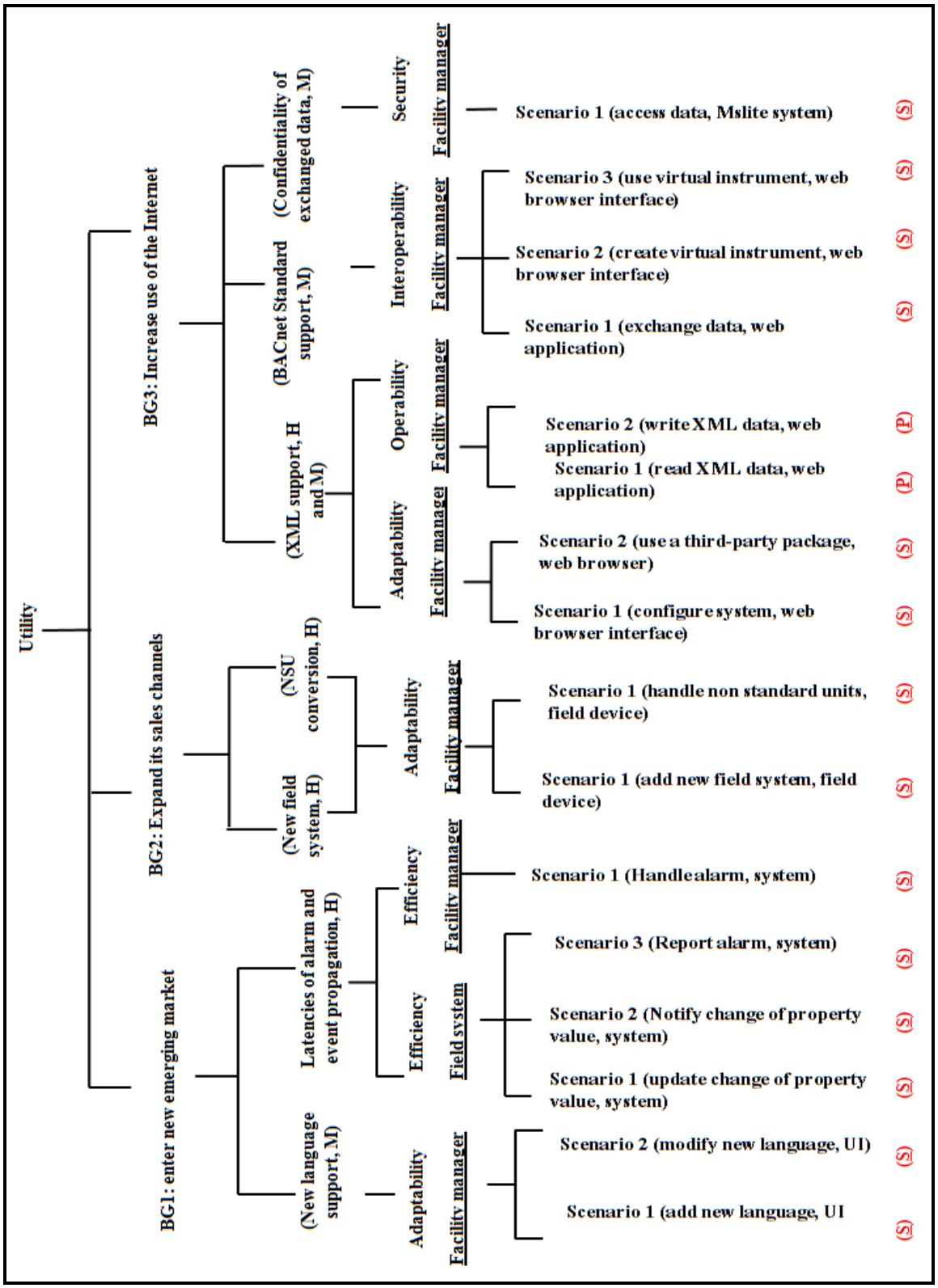


Figure 5.20 Utility tree with “Operability” partially satisfied

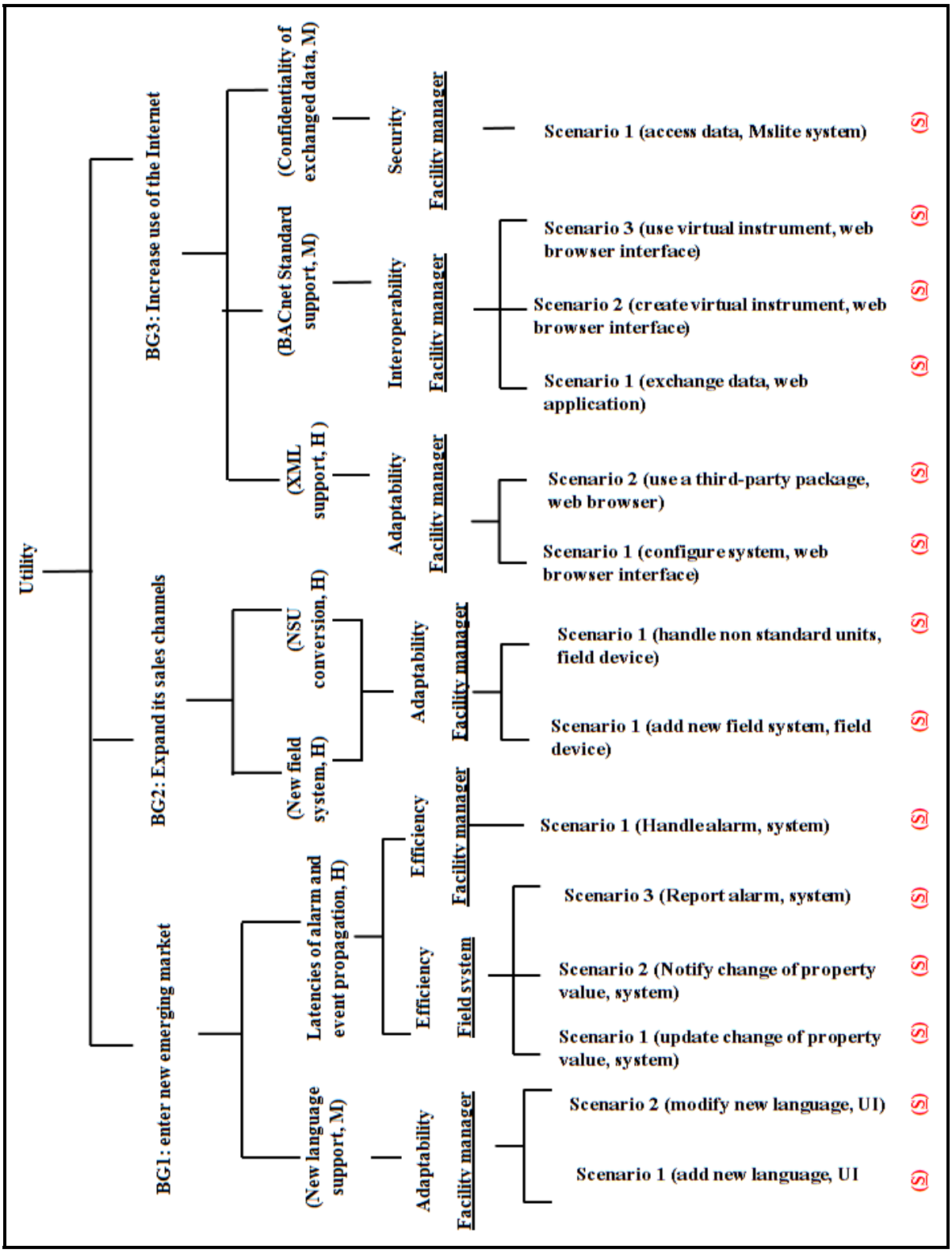


Figure 5.21 Consolidated utility tree

Table 5.13 Quality attributes template

Items	Description			
Name	Efficiency			
Description	Support regulations that require life-critical systems, such as fire alarms, to operate within specific latency constraints			
Category	External quality			
Source	Stakeholders, BMM and vision document			
Target stakeholders	Business manager, developer and evaluator			
Quality standard used	ISO/IEC Square 25030			
Priority	High for Business manager Medium for Developer High for Evaluator			
Representation	**BG1:**enter new emerging geographic markets:	**BG12:** latencies of alarm and event propagation	Field system 1. Scenario 1 (update change of property value, system) 2. Scenario 2 (Notify change of property value, system) 3. Scenario 3 (Report alarm, system	Facility manager 1. Scenario 1 (handle alarm, system)
Requirements	Functional requirements (described in the use case model)			
Activities and phases Standards	Architecture, testing ISO/IEC Square 25030 and ISO/IEC 14598			
Models and processes	Use case and business domain models			
Impact	(-) to Adaptability, (-) to Interoperability, (-) to Operability			

Phase 6: link quality attributes to functional requirements

The utility tree is mapped to use case and business domain models of MSLite system by using "**Mapping rules"** (Figure 5.22 with main concepts involved in phase 6).

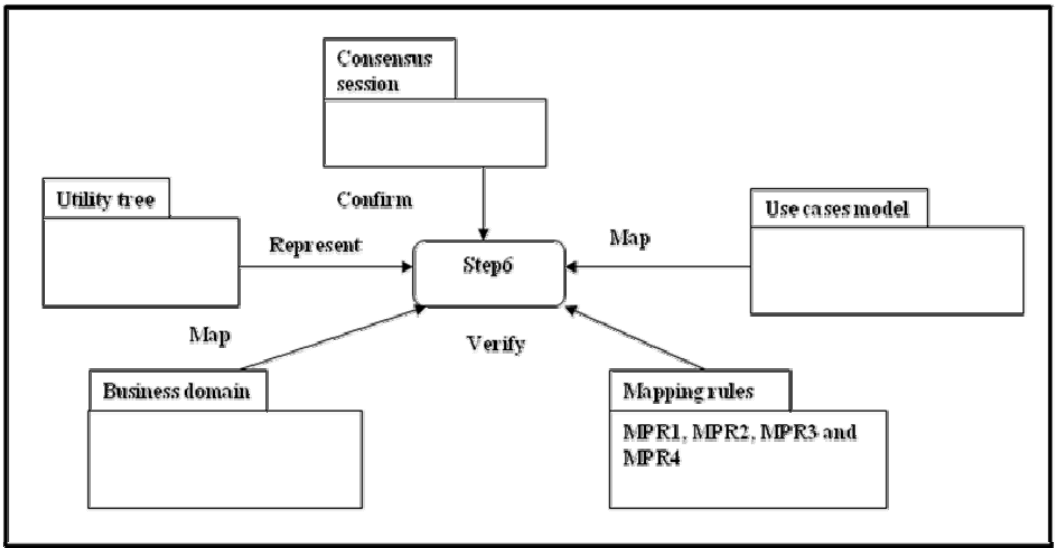


Figure 5.22 Concepts of phase 6

Quality attributes are linked to the functional requirements in two ways:

1. **By the use case model** (Figure 4.16): from the consolidated utility tree (Figure 5.21), map the actions of quality attributes scenarios to candidate use cases of the functional process. The original use case model (Figure 5.3) is adjusted to the new model (Figure 5.23). The new added uses cases are:

    a. Define language;
    b. Handle NS-units;
    c. Configure the system;
    d. Report life-critical alarms;
    e. Update change of value.

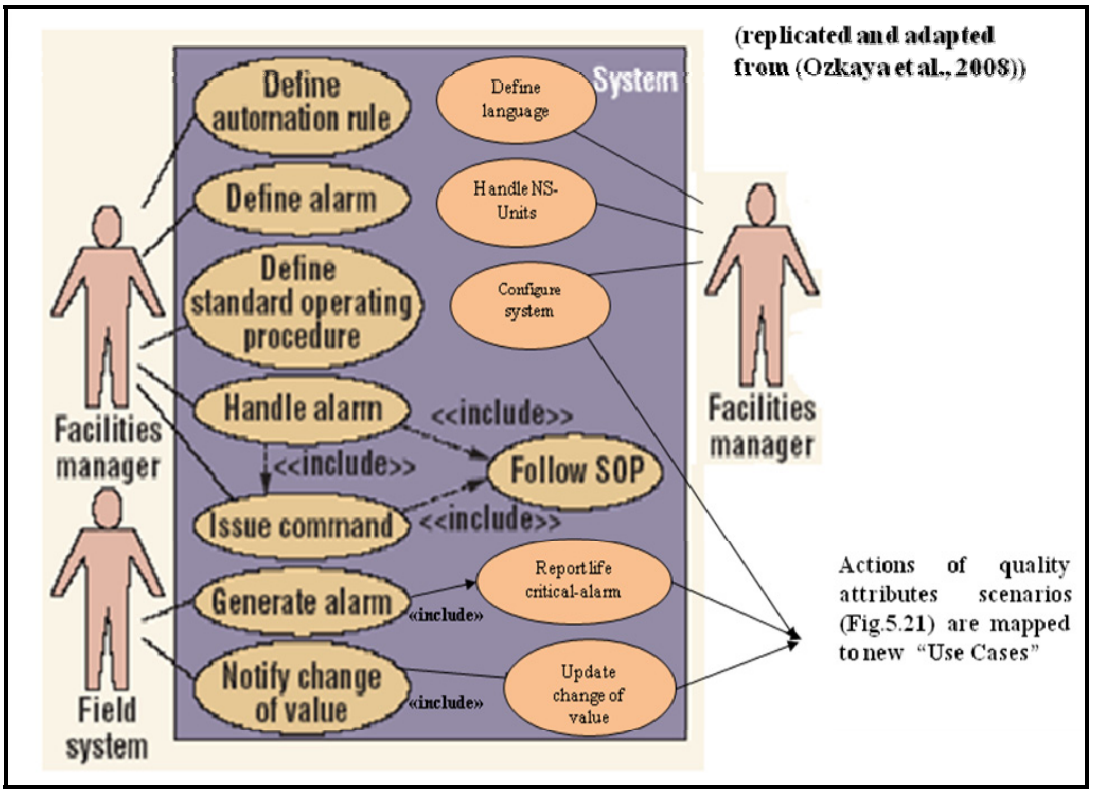


Figure 5.23 Extended use case model with Adaptability and Efficiency scenarios

2. **By the business domain model** (Fig 4.17): the business domain model of Figure 5.4 is extended with quality attributes concepts. The following actions are undertaken by the actor *facility manager* in the quality scenarios (Figure 5.21):

    1. Define field system language;
    2. Convert field system in new "NS units";
    3. Configure system with internet communications capability.

These actions are mapped to the following business concepts in the business domain model (Figure 5.24):

1. International language;
2. “NS-Units”;
3. Field system which already exists;
4. Web browser package.

Relationships of the *facility manager* with the mapped business concepts are:

1. Defines: between the *facility manager* with International language;
2. Converts: between the *facility manager* and the NS Units concept;
3. Configures: between the *facility manager* and the Web browser package.

Figure 5.24 shows the extended business domain model with business concepts.

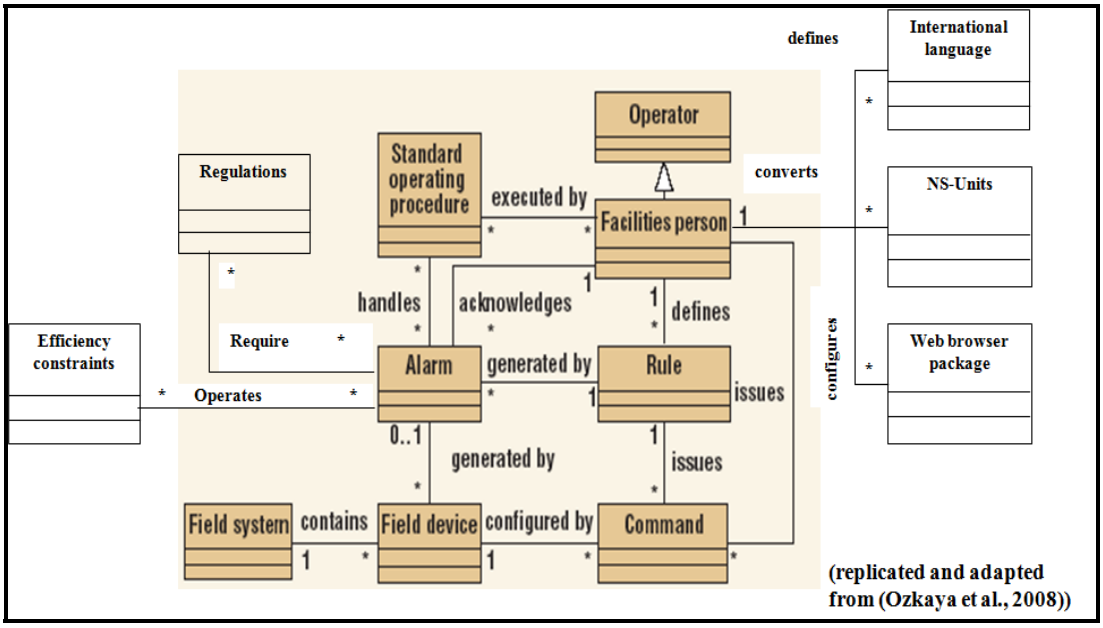


Figure 5.24 Extended business domain model with new business concepts

It is possible to define the “Adaptability” view which is projected from the added business and relationship concepts (Figures 4.18 and 5.25). For example, the “facility manager” concept is related to three added business concepts: “International language”, NS-Units and “Web browser package” by the three relationship concepts (defines, converts by and configures).

The second quality view to be projected from the added business concepts “constraints” and “regulations” is “Efficiency” view where business concept “Alarm” is related to business concepts “Regulations” and “Constraints” by the relationships concepts (require and operates). In fact, alarm is supported by regulations requiring its operation under certain latency constraints. Figure 5.25 shows the quality views (Adaptability and Efficiency) projected from the added business concepts (for the building automation system).

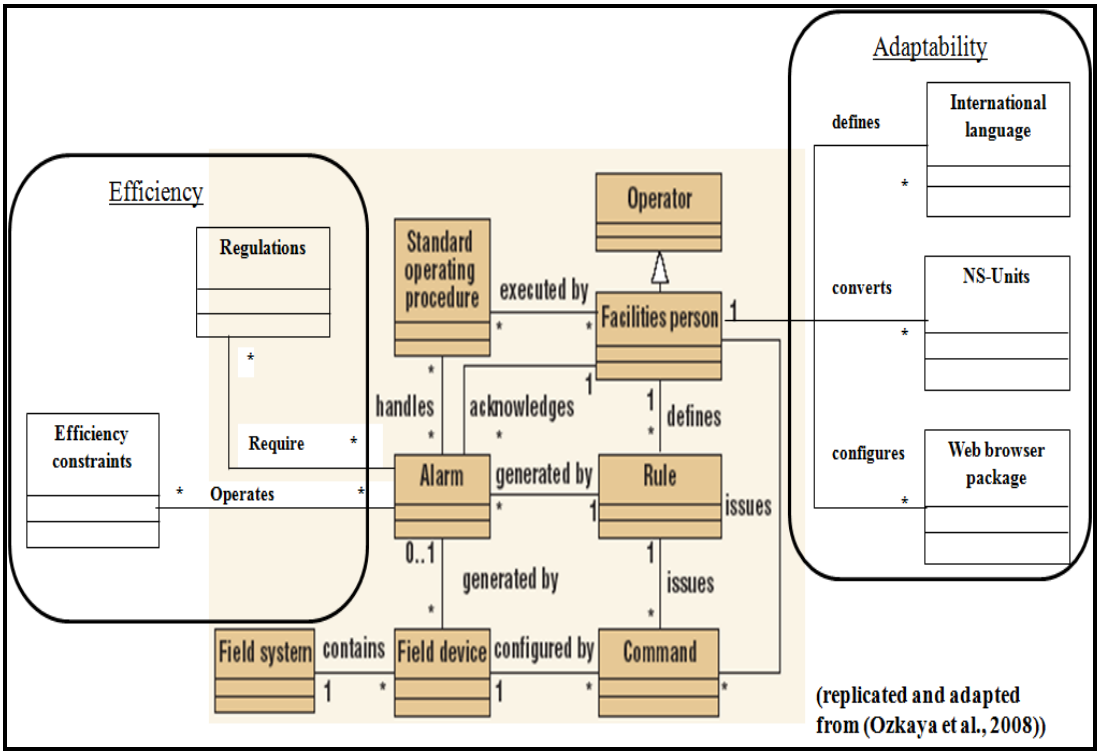


Figure 5.25 Quality views of new business concepts

5.1.4 Analysis of SOQUAREM process

SOQUAREM process was evaluated by professionals and experts from the requirements engineering community in order to know the extent of addressing quality requirements. The task was not easy since the evaluation of the whole process requires several steps difficult to perform during the mandate of this thesis. In this section, an evaluation step was suggested in two ways: a) by interviewing and obtaining feedback from participants of the ISSEM 2011 Workshop via a survey (Annex II-1); and b) by gathering feedback from international software quality experts (Annex II-2).

1. **Interviewing the participants of the workshop:** performed via a questionnaire conducted with quality engineering participants (PHD students, architects and practitioners) to get valuable feedback on applicability, appropriateness, understandability and completeness of SOQUAREM process (the four phases of the process have been evaluated and responses were collected from four participants (Annex II-2)). The objective was to identify to what extent SOQUAREM addressed the chosen quality requirements management activities: identification and representation. Hence, an evaluation was made to see if the used concept at each activity of the process was **adequately applied** (if the concept is applied in the example according to its definition), **appropriately used** (if the concept is applied in the right and corresponding place) in the example and **easily understandable** (the concept is applied in the example without much cognitive workload**)**. **Missing elements** (elements have been missed from the use of the concept) were acknowledged from the process related to these criteria. Questions were categorized into four major evaluation criteria:

   i. Applicability of SOQUAREM process (phases, concepts and techniques) in terms of identification, representation, traceability and documentation;
   ii. Appropriateness of the way SOQUAREM process used concepts and techniques;
   iii. Understandability of concepts during application of SOQUAREM process activity;
   iv. Completeness of SOQUAREM process according to the used concepts and techniques.

More specifically, SOQUAREM has been evaluated from different axes:

1. Activities of SOQUAREM and the used concepts at each activity and phase of its process (Annex II-1-2: Tables A II- 3 and A II-4);
2. Applicability of the method according to chosen criteria from literature (Annex II-1-2 and Table-A II- 7);
3. Dealing with software quality (Annex II-1-2 and Table-A II- 8);

1.1 Activities of SOQUAREM and the used concepts at each activity and phase of SOQUAREM

1.1.1 For the identification activity: collected responses from the participants are given for the four evaluation criteria: Applicability, Appropriateness, Understandability and Completeness: see Tables 5.14-5.17 and Figures 5.26-5.29. See also Annex II-1-2: Table-A II- 3 and Annex II-2.The following scale is used: 3 = very good concept; 2= fair concept and 1= poor concept.

Table 5.14 Applicability of concepts for the identification activity

Response no	Applicability of BMM	Applicability of BCT	Applicability of Scenarios template	Applicability of transformation rules	Applicability of ISO/IEC 9126
Response 1	3	3	1	3	3
Response 2	3	3	2	3	1
Response 3	2	2	3	3	1
Response 4	3	1	2	2	3

The scenario template is not a concept used in the identification activity during the first phases of the process: 1, 2 and 3. It has been introduced in this evaluation to know if participants have read and understood SOQUAREM process. One of the participants has identified this error (the participant 1 Annex II-2-1).

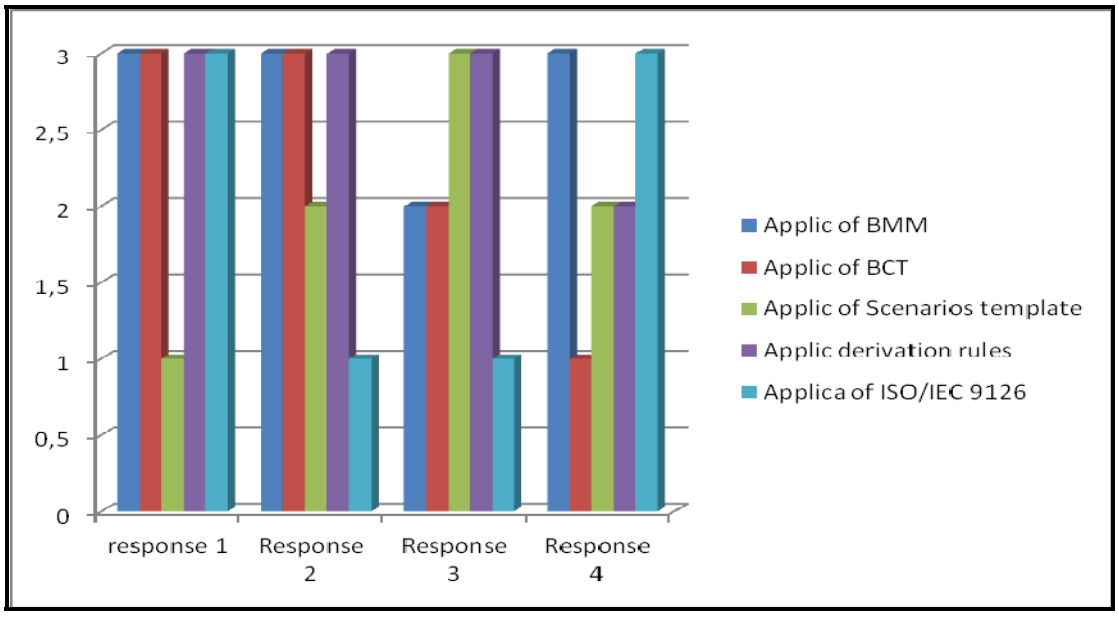


Figure 5.26 Responses of participants about applicability of concepts

One can say that BMM and transformation rules are the most adequately applied concepts. The BCT concept is in second position. The scenario template and the quality standard follow.

Table 5.15 Appropriateness of concepts for the identification activity

Response no	Appropriateness of BMM	Appropriateness of BCT	Appropriateness of Scenarios template	Appropriateness of transformation rules	Appropriateness of ISO/IEC 9126
Response1	3	2	1	3	3
Response	3	3	2	3	1
Response 3	2	2	3	3	2
Response 4	1	3	2	1	3

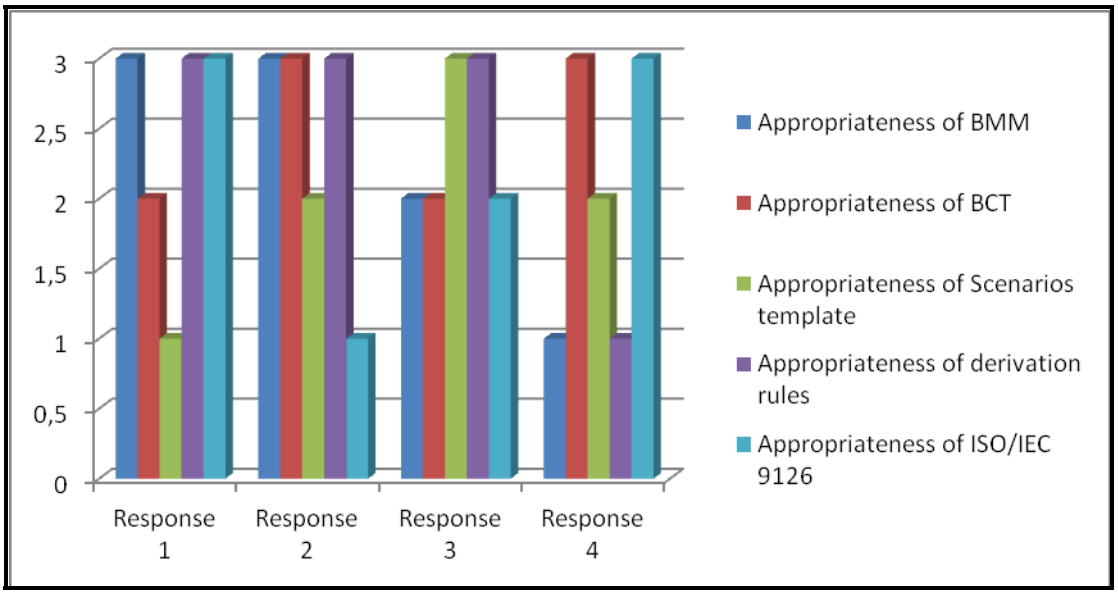


Figure 5.27 Responses of participants about appropriateness of concepts

The transformation rules and BCT concepts are the most appropriately used, followed by the BMM and ISO/IIEC 9126 quality standard. Finally, the scenario template is in third position.

Table 5.16 Understandability of concepts for the identification activity

Response no	Understandability of BMM	Understandability of BCT	Understandability of Scenarios tempalte	Understandability of transformation rules	Understandability of ISO/IEC 9126
Response 1	3	2	1	3	2
Response 2	3	3	1	3	1
Response 3	2	2	2	2	1
Response 4	3	3	2	2	2

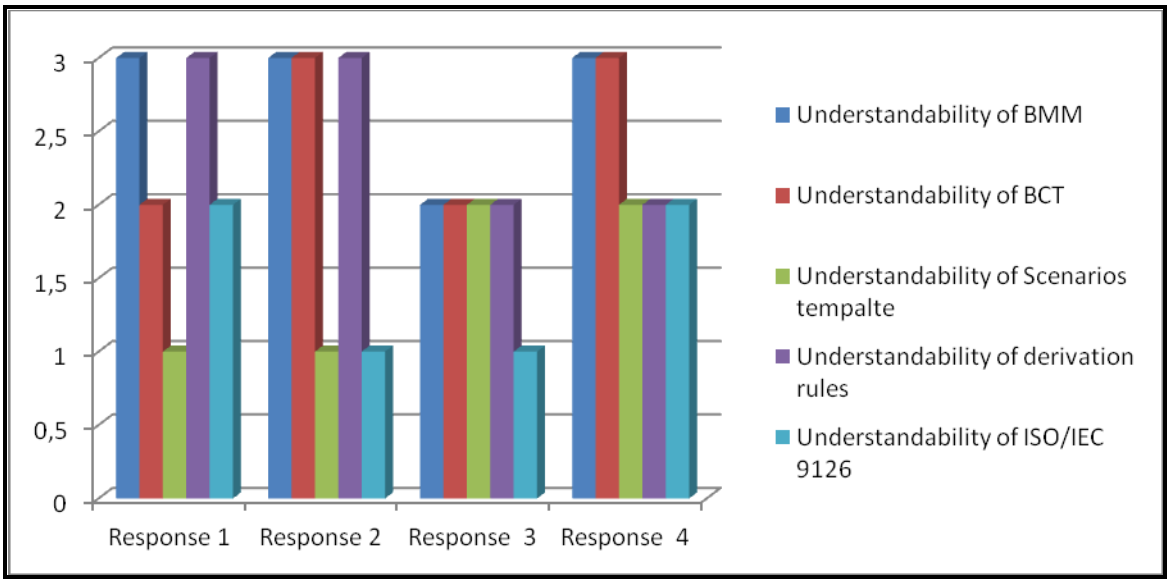


Figure 5.28 Responses of participants about understandability of concepts

BMM is an easily undertsandable concept, then come the BCT and transformation rules. Finally, the scenario template and the quality standard are in third position.

Table 5.17 Completeness of concepts for the identification activity

Response no	Completeness of BMM	Completeness of BCT	Completeness of scenarios template	Completeness of transformation rules	Completeness of ISO/IEC 9126
Response 1	2	3	1	3	3
Response 2	3	3	1	3	1
Response 3	2	2	3	3	2
Response 4	2	3	3	2	3

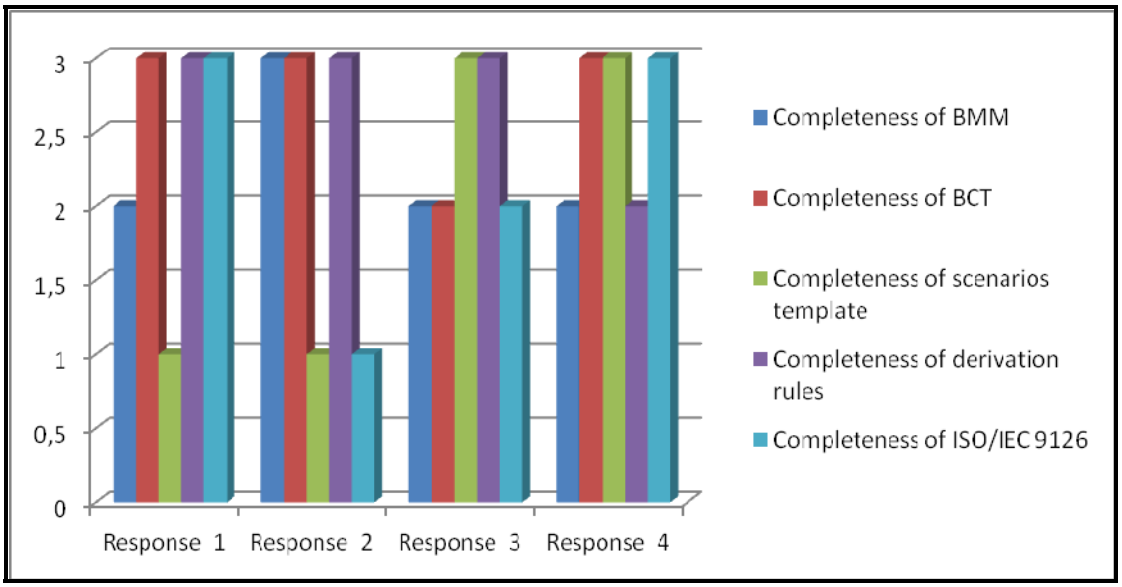


Figure 5.29 Responses of participants about completeness of concepts

The BCT and transformation rules have no major missing elements. In second position, BMM and ISO/IEC 9126 quality standard and finally the scenario template is quoted in third position.

From collected responses, one can say that SOQUAREM addresses well the identification of QAs by the following concepts:

- BMM, “Transformation rules” and BCT are the most adequately applied, and easily understandable concepts;
- BCT, “Transformation rules”, BMM and ISO/IEC 9126 are the most appropriately used concepts. They are also applied without major missing elements that contribute to identify the QAs;
- The scenario template concept was not well ranked by participants for the identification activity because it is not used during phases 1, 2 and 3 of SOQUAREM process.

1.1.2 For the representation activity: collected responses from the participants are given for the four evaluation criteria: applicability, appropriateness, understandability and completeness: see Tables 5.18-5.21 and Figures 5.30-5.33. See also Annex II-1-2: Table-A II- 4 and Annex II-2.

Table 5.18 Applicability of concepts for the representation activity

Response no	Applicability of Utility tree	Applicability of Scenarios template
Response 1	3	1
Response 2	3	2
Response 3	3	3
Response 4	3	2

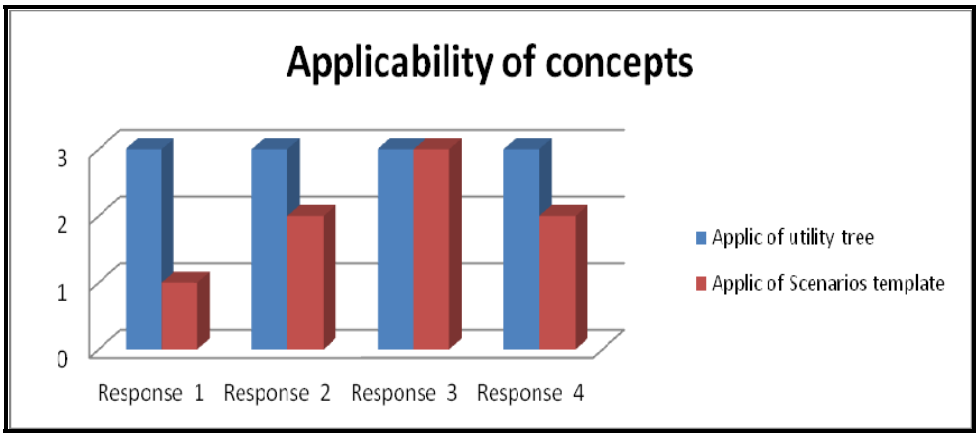


Figure 5.30 Applicability of concepts

Table 5.19 Appropriateness of concepts for the representation activity

Response no	Appropriateness of Utility tree	Appropriateness of Scenarios template
Response 1	2	1
Response 2	3	2
Response 3	3	3
Response 4	3	3

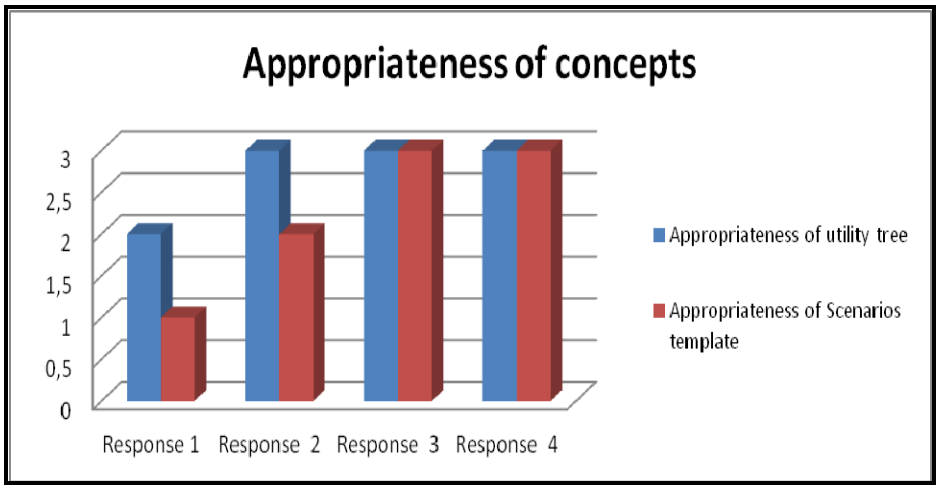


Figure 5.31 Appropriateness of concepts

Table 5.20 Understandability of concepts for the representation activity

Response no	Understandability of Utility tree	Understandability of Scenarios template
Response 1	3	1
Response 2	3	1
Response 3	3	2
Response 4	3	3

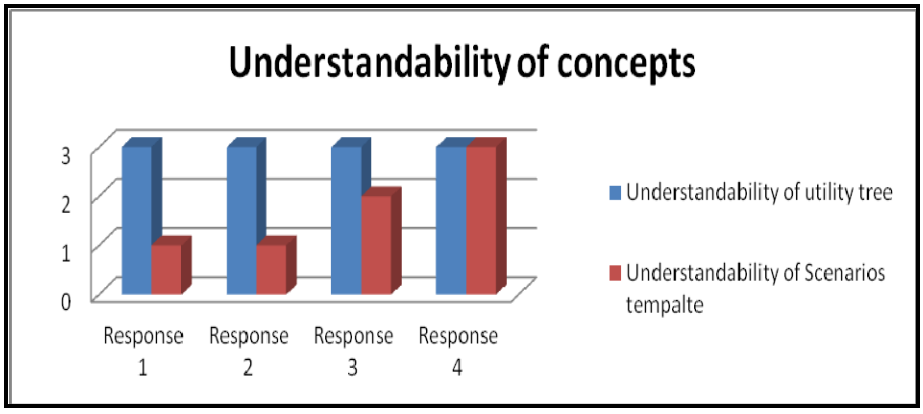


Figure 5.32 Understandability of concepts

Table 5.21 Completeness of concepts for the representation activity

Response no	Completeness of utility tree	Completeness of scenarios template
Response 1	2	1
Response 2	3	1
Response 3	3	2
Response 4	3	2

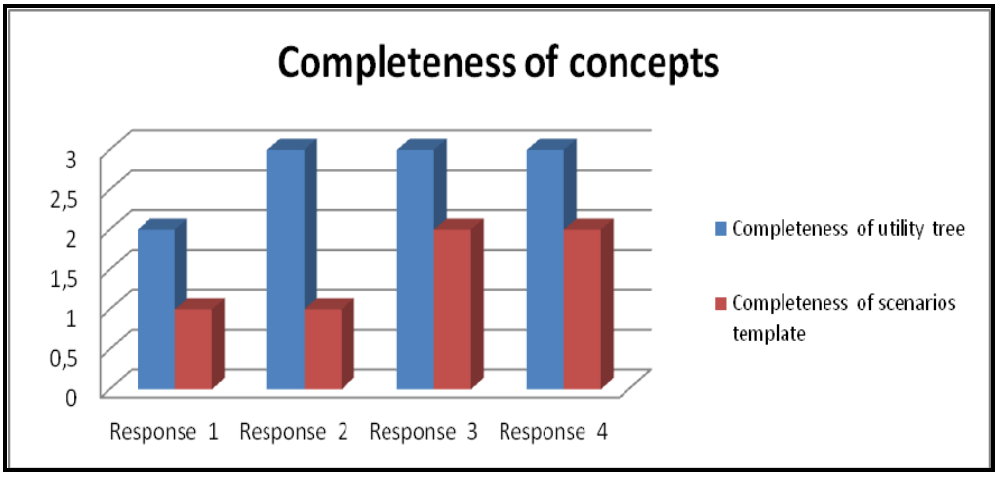


Figure 5.33 Completeness of concepts

From collected responses, one can say that SOQUAREM addresses well the representation of QAs by its involved concepts. Compared to the scenario template concept, the utility tree is the most adequately applied, easily understandable and appropriately used concept. It is also used without any missing elements that contribute to represent QRs.

In this section, one observes that:

- SOQUAREM is able to identify quality attributes by its used concepts like BMM, BCT and “Transformation rules”;
- SOQUAREM is also able to represent quality attributes by its used concepts like utility tree and scenario template;
- Concepts are adequately applied, appropriately used in the example and easily understandable. Concepts are also used without major missing elements that could contribute to represent the QAs;
- The scenario template has been evaluated as a very good concept (applicability, appropriateness and understandability) for phase 4 of SOQUAREM process (Annex II-2-1);
- The utility tree concept describes the traceability between quality attributes and business goals very well.

1.2 Applicability of the method according to chosen criteria from literature

Collected responses are illustrated in Tables 5.22 and Figures 5. 34. See also Annex II-1-2: Table-A II-7 and Annex II -2.

Table 5.22 Applicability of SOQUAREM

Response no	Adaptability to QRs	Client acceptance	Complexity	Scalability
Response 1	3	3	3	3
Response 2	3	2	2	2
Response 3	3	1	2	3
Response 4	3	3	2	3

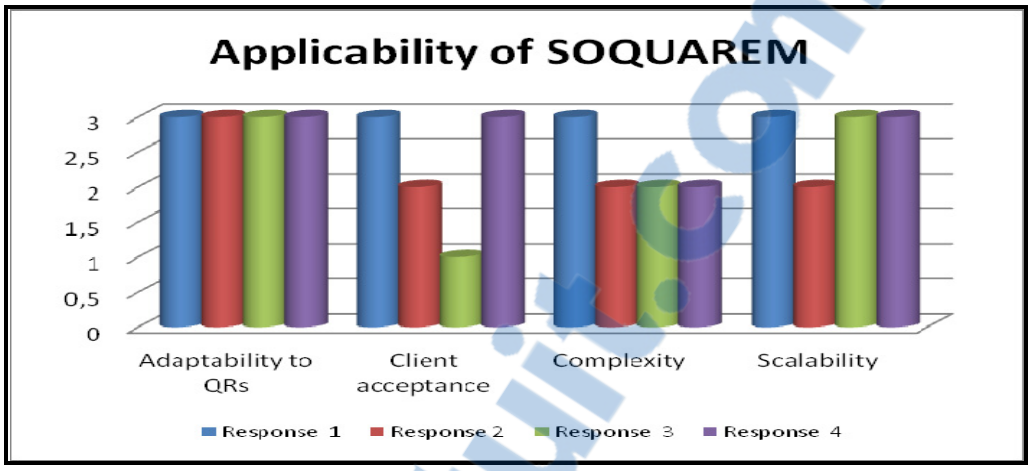


Figure 5.34 Criteria for applicability of SOQUAREM

SOQUAREM has been evaluated according to four criteria: adaptability to QRs; client acceptance; complexity and scalability. “Adaptability and “Scalability” have been well accepted by evaluators. For “complexity“, the method is simple to understand but requires time. For the “client acceptance”, one notes that clients never have time to read SOQUAREM and its use will demand time. In addition, it is important to tell customers why QAs are important.

1.3 Dealing with software quality

This section collected responses from participants about ability of SOQUAREM to deal with software quality and how it is possible to improve its process, challenges and further comments (Annex II-1-2: Table-A II-8 and Annex II -2). They are presented as follows:

1. SOQUAREM allows one to:
   - Represent in a very structured and simplified way all the relevant concepts;
   - Trace back to the high level needs that caused each QA;
   - Prioritize the development effort by QAs requirements management and change request;
   - Easily understand its use but it is time consuming because managers do not have the time to define goals clearly;

2. SOQUAREM will be improved by developing an automated tool and measuring its benefits from real cases;
3. Strengths of the method are: utility tree concept, very structured methodology, easy to apply (taking one hour (Annex II.2.3 section “Other findings”));
4. It could be used in an academic environment to introduce to students;
5. Threats to SOQUAREM are: how it links to agile methods (Annex II.2.3 section “Other findings”);
6. It is important to market the method and link it to a QAs model and measurement process;
7. It is also suggested to market the present advantages of QAs requirements management;
8. The challenges of applying SOQUAREM are foreseen in costs and time for gathering requirements. It also requires an additional effort from the requirements professional to encourage some companies to be more involved in the software QRs management process.

2. **Feedback from the potential experts in the software quality field:** obtain valuable feedback from experts in the quality software engineering field. SOQUAREM has been submitted and revised by international software quality experts (A list of experts is presented in Annex II-2, only three experts have given their feedback). Results collected from the experts revealed the following points:

    1. SOQUAREM is well structured and easy to read;
    2. The concepts used in SOQUAREM could improve derivation of quality attributes from business goals especially BMM and BCT concepts;
    3. The QAs template needs some improvements:
        a. Representation of the quality scenarios and their prioritization;
        b. Detail more activities of the software life cycle and phases of the software process standards.
    4. Resolving the conflicting attributes is very important and difficult to implement.

5.2 Conclusion

The present chapter described in an illustrative example the application of SOQUAREM to the MSLite system, a unified management station for the building automation domain.
At first, the main inputs/outputs of SOQUAREM process were defined, followed by a detailed description of the application of SOQUAREM phases to MSLite system.

Later, SOQUAREM method was analyzed and evaluated in two ways:

1. Interviewing experts of the ISSEM 2011 workshop via a survey and collecting their feedback;
2. Gathering feedback of the international software quality domain experts.
   Analysis revealed the importance of a structured and easy to use process by practitioners. Results also show the valuable contribution of used concepts such as: BMM, BCT, utility tree and scenario template in the management of quality requirements (identification, representation and traceability).

The next chapter concludes this thesis document

CONCLUSION

A. Summary of investigations

This thesis presented and described a software quality requirements engineering method called SOQUAREM (SOftware QUAlity Requirements Engineering Method). Its main objective is to support identification and representation of quality requirements at the definition phase of a software product. SOQUAREM is born from the ideas of: a) the motivation of the business which contributes to align business specifications to system and user requirements; b) supporting QRs management techniques by quality standards; c) providing clear and structured guidance on how to elicit, document and retrace QRs and d) integrating the software QRs specifications into the functional process. It provides a general conceptual model which derives quality attributes from business goals and ensures their properly detailed definition. SOQUAREM addressed the challenging aspects of software QRs management such as identification (of business and software) requirements, conflicts resolution and prioritization, representation and traceability, specification and documentation of QRs. Dedicated to address all types of quality requirements, SOQUAREM provides structured engineering process phases supported by the ISO/IEC 25030 standard and concepts of different organizational levels to systematically define and represent quality requirements.

The conceptual model of the method has been detailed including:

1. Business concepts such as BMM (Business Motivation Model) and BCT (Business Context Table);
2. Transformation rules (statement, refinement and linkage) to identify and derive important quality attributes according to ISO/IEC 25030 taxonomy;
3. Scenarios concept to infer the right quality attribute;
4. Utility tree to retrace quality attributes to their original requirements;
5. QAs template to specify and document quality attributes;

6. Mapping rules to integrate quality attributes into the functional model;
7. Finally, consensus sessions used at each process phase to interact with stakeholders and domain experts.

The different phases of the software QRs engineering process of SOQUAREM are described in the following levels:

- **Business level:** where business goals are identified and refined according to business context elements (phases 1 and 2 of the process);
- **System level:** where:

  a. The business goals are used to derive and infer the right QAs by using ISO/IEC 25030 quality standard, linkage rules and scenario template concepts (phases 3 and 4 of the process);
  b. QAs are analyzed for conflicts and consolidated according to the prioritized methods (impact matrix and weighted method) and the consensus sessions to select and confirm the most suitable QAs (phase 5 of the process);
  c. QAs are mapped to the functional requirements process by using the mapping rules and scenario template concepts. The initial use case model is updated with additional information about QAs **(**phase 6 of the process).

SOQUAREM concepts have been applied and illustrated in an example: “a building automation system” and a management station system called MSLite. In this example, SOQUAREM process has been applied to the MSLite system to deal with its high level technological constraints and meet the associated high level quality needs. Inputs to SOQUAREM process are: functional requirements (FRs), the BMM and BCT concepts and the main output were the list of quality attributes for MSLite system which were integrated in the use case and business domain models of the organization.

The main purpose of the example is to show that it is possible to manage QRs by performing subsequent refinement phases and verification rules from the abstract business goals to the detailed quality attributes.

SOQUAREM has shown its merits in this example and relevant feedback from international software quality domain experts and participants of the workshop ISSEM 2011 (section 5.1.4) demonstrated success points of this method.

B. Key Contributions

This research created a quality requirements engineering method for software product systems. The major contribution is the creation of the first structured quality requirements engineering process which:

1. Is designed from the foundations of the quality engineering standard ISO/IEC SQuaRE 25030;
2. Describes fully the derivation of quality attributes from business goals;
3. Integrates intuitive modeling and motivation of the business in the quality process in order to:
    a. align business specifications with system requirements and architectural design;
    b. derive and define quality attributes from business context elements;
    c. build the bridge between business and system level specifications;
4. Provides more interaction with stakeholders and domain experts during consensus and free dialogue sessions;
5. Integrates many concepts of recognized methods and standards to adequately manage software QRs such as utility tree of ATAM method and BMM standard of OMG;
6. Allows the integration of quality requirements into the functional process;
7. Integrates scenarios at the requirements level to help resolve terminology problems and infer the correct quality attributes;

Various contributions documented in this thesis have been published at conferences and in journals. The list follows:

Conference Papers

Published

1. Djouab R., Suryn W. (2006) “An ISO/IEC standards-based quality requirement definition approach: Applicative analysis of three quality requirements definition methods”. ISIE 2006 Annual Conference of the IEEE Industrial Electronics Society. 9-13 July 2006. Page (s): 3231 - 3239. Montreal, Que.
2. Djouab R., Suryn W. (2007) “Analysis of a probabilistic quality method for evaluation of non functional requirements” was published for ICSSEA International Conference on Software and Systems Engineering and their Applications. 4-6 December 2007 - Conservatoire National des Arts et Métiers - Paris, France
3. Djouab R., Suryn W. (2007) “Applicability analysis of two quality requirements treatment methods: IESE NFR and FDAF” was published for ICSSEA International Conference on Software and Systems Engineering and their Applications.4-6 December 2007 - Conservatoire National des Arts et Métiers - Paris, France
4. Djouab R., Suryn W. (2011) SOQUAREM: SOftware QUALity Requirements Engineering Method was published for SQM Conference on Quality Management. 18-20 April 2011. Loughborough University, Leicestershire, UK.
5. Djouab R., Suryn W. (2011) Applicability of SOQUAREM method: "an illustrative case study” was published for SQM Conference on Quality Management. 18-20 April 2011. Loughborough University, Leicestershire, UK.

Journal Papers

Submitted

1. Djouab R., Suryn W. (2012) "The bridge between business and system level specifications: SOftware QUAlity Requirements Engineering Method (SOQUAREM)". **RE 2012** journal. Reference number: RE388.
2. Djouab R., Suryn W. (2012) "Analysis and improvement of the IESE NFR method". **RE 2012 journal.** Reference number: RE389.
3. Djouab R., Suryn W. (2012) "How could BMM and GQM contribute together to capture quality attributes for the software product?" **RE 2012 journal.** Reference number: RE391.

C. Implications for software engineering theory

The quality requirements process opens a new research avenue to the development and management of quality requirements at early stages of development (requirements and design process). Once published, it will be a good enrichment for SWEBOK with software QRs base knowledge and will provide benefits to International Standards Organization ISO/IEC SC7/WG6.

Compared to existing quality methods, this research introduced:

1. Novel quality requirements engineering process called SOQUAREM for product software (chapter 4.2);
2. Novel quality engineering concepts including scenario template and transformation rules (statement, refinement, linkage and mapping rules) (section 4.1.2).

D. Practical implications

The results of this research have practical implications for the software engineering community. The proposed SOQUAREM method will offer to industry the facility to manage software QRs (by using a structured QRs process) to obtain the product software systems with the desired quality attributes that conform to the most recognized software quality engineering standards (ISO/IEC SQuaRE 25000).

The use of various concepts will offer industry a flexible model to understand management of software QRs and how to deal with them appropriately. Application of the business concepts BMM and BCT will provide a better understanding of the motivation of business. Scenario template will allow understanding the purpose of each quality attribute while the utility tree will provide an easy way to retrace quality requirements to their original business requirements. Moreover, consensus sessions will improve communication between quality practitioners and stakeholders. Alignment between business and functional requirements will facilitate the specification of architectural styles and increase mutual understanding between software architects, business managers and quality practitioners.

E. Limitations and strengths

SOQUAREM has been developed to support quality practitioners and software engineers in identifying and representing quality attributes of the software product. It is easy to apply but requires time and effort to become familiar to interested stakeholders. SOQUAREM process also offers to stakeholders an opportunity to learn more about QAs and to integrate them in their business process.

Current limitations of SOQUAREM include:

- A need to develop more the transformation rules (section 4.2.1.6) and the QAs database (section 4.2.1.3);

- SOQUAREM process has been evaluated partially and there is a need to evaluate the whole process with more standards/methods;
- SOQUAREM needs also to be evaluated in a real case (industrial context);
- There is a need for a supporting software tool to better improve communication among interested stakeholders;
- There is a need for the most recognized prioritization methods and automated modeling systems for conflict resolution support like AHP (Analytical Hierarchy Process) and S-COST (Software Cost Option Strategy Tool);
- SOQUAREM does not define measures for the defined QAs;
- There is a need for the academic environment to support to introduce SOQUAREM to students and the scientific community in order to contribute to the design of quality processes in organizations;

Further research is required to address these limitations, one by one.

The strengths of SOQUAREM method are:

- SOQUAREM process could be easily used by beginners as well as experts;
- SOQUAREM supports identification of QRs at early stages of the software life cycle;
- SOQUAREM supports communication and increases mutual understanding among stakeholders;
- SOQUAREM supports integration of QAs into the functional process;
- SOQUAREM supports alignment of business specifications with functional requirements.

F. Further research

Possible continuation of this research includes:

1. Application and validation of SOQUAREM in an industrial context;
2. Integration of the measures in the SOQUAREM process according to the updates of ISO/IEC 25030;
3. Development of a supporting IT tool that automates the SOQUAREM process and shows relevant parts of SOQUAREM process model;
4. More development of the mapping between the ISO/IEC 25030 concepts and the SOQUAREM process;
5. More rework and development of the transformation rules;
6. Further work on the mapping rules and the integration process of the QAs into the FRs process;
7. Evaluation of the applicability of SOQUAREM process through applying appropriate ISO/IEC standards;
8. Supporting SOQUAREM process with prioritization methods and automated modeling systems like AHP (Analytical Hierarchy Process) and S-COST (Software Cost Option Strategy Tool);
9. Deployment of the questionnaire on QRs engineering practices in a large industrial spectrum;
10. Integration of SOQUAREM process with software engineering processes/methods such as agile methods, RUP (Rational Unified Process), RAD (Role Activity Diagramming), Architecture centred design and ATAM (Architectural Trade Off and Analysis Method).

ANNEX I

QUESTIONNAIRE ON QRS OF THE SOFTWARE PRODUCT

Annex I is divided into two parts. The first part describes the questionnaire and its sections. The second presents the row data collected from the interviewed experts.

I.1 Description of the questionnaire

This part describes the purpose of the questionnaire and its important sections

I.1.1 Purpose of the questionnaire

Dear Sir / Madam

We are studying quality requirements for software, particularly quality requirements in the software development life cycle.

This survey is aimed at identifying quality requirements used in industry which will help identify critical needs in this field, as well as the difficulties faced with their processing.

Results of the survey will be useful for identifying the best software engineering practices in use. Your contribution is important to the success of our research objective. Gathered data will remain confidential and all data will be made anonymous.

We thank you in advance for your participation as part of our research.

This survey starts with a series of questions regarding the person completing the survey. This includes information such as the number of years of experience in the field of quality requirements for software.

Next, a series of questions related to processes, methods, software quality engineering standards and stakeholders interested in quality requirements.

Please answer questions based on your experience in the field of software product quality. Additional information may be written in the space provided.
It should take nor more than 30 minutes to answer this survey.

Participants may have a copy of any findings if they desire. If you have questions about the follow up of the survey or you have any concerns about the research, please do not hesitate to contact me at my email address below. If you want to see a summary of the results, I can send them at the end of August.

Thank you

Rachida Djouab
Ph.D. Eng.Student
==========
Dept of SW and IT Engineering
École de Technologie Supérieure – ÉTS
1100 Notre-Dame Ouest
Montréal, Québec, Canada H3C 1K3
rachida.djouab.1@ens.etsmtl.ca; rdjouab@hotmail.com

I.1.2 Description of sections on the questionnaire

Sections of the questionnaire are listed as follows:

1. Identification of the respondents (Table- A I-1);
2. Companies and stakeholders (Table- A I-2);
3. Processes (Table- A I-3);
4. Methods (Table- A I-4);
5. Standards (Table- A I-5).

Table- A I-1 List of domain experts

FORM Of IDENTIFICATION OF THE RESPONDENTS	
Family Name: _____ Forename (s): _____ Date (dd/ mm /yyyy: ____ / ____ / _____27 September 2012	
Questions	Answers
1. What is your position within your organization nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Quality engineer ☐ Quality assurance manager ☐ Others, specify _____ ☐ Comments
2. How long have you been worked in the area of software quality requirements (in years)?	☐ Less than 1 year ☐ 1-3 years ☐ Over 3 years ☐ Others, specify_____ ☐ Comments
3. What are your responsibilities? 1.	☐ Planning of software ☐ Design of software ☐ Specification of software ☐ Programming and test ☐ Construction or Installation ☐ Maintenance of software ☐ Test of software ☐ Others , specify_____ ☐ Comments

Table- A I-2 Companies and stakeholders

COMPANIES AND STAKEHOLDERS SECTION	
This section collects information on activity fields of companies and the important stakeholders that can be interested by the processing of quality requirements of the software product.	
Questions	Answers
1. Are you working for:	□ a small part of a bigger company □ a small part of a smaller company □ a big part of a bigger company □ a big part of a smaller company □Other, please specify_____ □ Comments _____ _____
2. What does your company do? Please specify the following activity domains? For example: For the largest part, add 1 asterisk next to the activity domain *□ For the most important part, add 2 asterisks **□ For the relevant ones, add 3 asterisks, ***□	2. □Aeronautics □Electronics □Banking □Education □Research and development □Health □Security □Others, specify _____ □ Comments _____ _____
3. What types of projects and software are developed by the company? Please specify the relevant ones. For the most important, please add an asterisk next to them?	□Systems □Real time □Business □Scientists □Embedded □Personal □Internet based □Others, specify _____ □ Comments _____ _____
4. What is the business critical level of your software products? (For example if the software product is related to critical systems such as fire systems and nuclear systems).	□critical □no critical □Others, specify _____ □ Comments _____ _____

COMPANIES AND STAKEHOLDERS SECTION

5. Who are the stakeholders interested in processing of quality requirements?

- Department of software development?
- Department of IT or business operations?
- Department of management
- Department of marketing
- Department of sales
- Others, specify _____
- Comments _____

6. Who is responsible for managing quality requirements for a specific software development project?" Please, specify his experience?

Quality responsible	Experience (years)
Project administrator Project manager Software or systems developer Quality engineer Others, specify _____	Under 1 year 1 year and more Others, specify _____

- Comments _____

7. Please, specify the type of training given in software quality for the several people?

	Norms and standards	Processes and methods	Software tools	Others
Project administrator				
Project manager				
Software or systems developer				
Quality engineer				
Others, specify:_____				

- Comments _____

Table- A I-3 Processes

SECTION ON PROCESSES	
This section collects information on the existent processes dealing with quality requirements of the software product	
Questions	Answers
1. In your organization, do you use a quality requirements process? (Identification, specification, representation, documentation and prioritization or others)? Please specify the most accurate answer?	1. Use all the time 2. Use sometimes 3. Have done but stopped 4. Plan to but not yet 5. No plans to ☐ Comments: _____ _____
2. If your organization uses a quality requirements process, please specify which of the following activities are performed by your process? If there are most important activities, please add an asterisk next to them?	□Identification of quality requirements □Specification of quality requirements □Representation of quality requirements □Prioritization of quality requirements □Documentation of quality requirements □Comments _____ _____
3. Is the quality requirements process supported by software tools?	□From Rationale software Inc. □Internally developed □No, we do not use a software tool □Comments _____
4. Could a structured and well defined quality requirements process improve the quality of your projects? Please provide your personal opinion.	□ Yes [for example, we need to identify quality attributes, represent them and document them] _____ □ No_____ _____ □ N/A_____ □ Comments: _____ _____

Table- A I-4 Methods

<table>
<tr><th colspan="2">SECTION ON METHODS</th></tr>
<tr><td colspan="2">This section collects information on the existent methods of quality requirements processing of the software product</td></tr>
<tr><th>Questions</th><th>Answers</th></tr>
<tr><td>1. In your organization, are quality requirements identified according to specific methods (for example using interviews with stakeholders to identify most quality attributes of the software product)/ techniques (or use checklists to elicit and document them)?</td><td>□ Questionnaire
□ Brainstorming
□ Observations
□ Meetings
□ Interviews
□ Checklists,
□ Internal methods of organizationNo, we do not use any identification method
□ Comments
_____</td></tr>
<tr><td>2. In your organization, do you use a specific method to decompose quality requirements into quality attributes?</td><td>□ Quality model
_____
□ Tree
_____
□ Graphical notation
_____
□ No, we do not use any decomposition method
□ Comments _____</td></tr>
<tr><td>3. In your organization, are quality requirements documented according to a definite formalism?</td><td>□In a requirements specification document (RSD)_____
□In Template_____
□ No, we do not use any documentation formalism
□ Comments _____</td></tr>
<tr><td>If your organization deals with quality requirements:

4. Could you specify the size of your software projects (SWPs)? KLOC (Kilo or thousands Lines of Code), Please specify the most important ones?</td><td>
<table>
<tr><th></th><th>Small</th><th>Medium</th><th>Big</th><th>Mega</th></tr>
<tr><td>Size of SWP1</td><td>< 50 KLOC</td><td>50-300 KLOC</td><td>300-1M. KLOC</td><td>>1M. KLOC</td></tr>
<tr><td>Size of SWP2</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Size of SWP3</td><td></td><td></td><td></td><td></td></tr>
</table>
</td></tr>
</table>

SECTION ON METHODS

This section collects information on the existent methods of quality requirements processing of the software product

☐ Comments

_____

_____

_____

5. Could you specify the total effort for the software projects (SWPs) (e.g. three people for two weeks, one person for 10 weeks)?

	Small	Medium	Big	Mega
Effort of SWP1	Person ☐ Day ☐ Week ☐	Person ☐ Week ☐ Month ☐	Person ☐ Month ☐ Year ☐	Person ☐ Year ☐
Effort of SWP2	Person ☐ Day ☐ Week ☐	Person ☐ Week ☐ Month ☐	Person ☐ Month ☐ Year ☐	Person ☐ Year ☐
Effort of SWP3	Person ☐ Day ☐ Week ☐	Person ☐ Week ☐ Month ☐	Person ☐ Month ☐ Year ☐	Person ☐ Year ☐

☐ Comments

_____

_____

_____

6. Could you specify the hierarchy of levels of authority of your software projects (SWPs)?

	Small	Medium	Big	Mega
Hierarchy of level of SWP1	1 level	2 levels	> 2 levels	>>
Hierarchy of level of SWP2				
Hierarchy of level of SWP3				

☐ Comments

_____

_____

_____

SECTION ON METHODS

This section collects information on the existent methods of quality requirements processing of the software product

7. Could you specify the duration of your software project?

	Small	Medium	Big	Mega
Time for SWP1	< 2 years	2-3 years	3-5 years	>5 years …
Time for SWP1				
Time for SWP1				

□ Comments

_____

_____

Table- A I-5 Standards

SECTION ON STANDARDS	
This section collects information on the software quality engineering standards of the software product used in industry	
Questions	Answers
1. In your organization, is the quality requirements process supported by a quality standard?	□□ISO / IEC 9126_____ _____ □□ISO / IEC 14598_____ _____ □ IEEE Std 830_____ _____ □ Comments _____
If your organization uses a quality standard, please specify? 2. Who is responsible for the application of this standard and for how long have they been responsible?	Responsible for standard: □□Project administrator; □□Project manager; □□Developer; □□Engineer quality; □□Quality assurance manager; □□Others, specify How long (years) □ Comments _____
If your organization uses ISO / IEC 9126 quality standard, please specify? 3. Used parts of this standard? 4. Frequency of their utilization (by number of Projects)?	Parts of ISO / IEC 9126: □□ Quality Model; □□Internal Quality; □□External Quality; □□Quality in use; □□Others, specify frequency (number of times by number of projects) □ Comments _____
5. If your organization does not use a quality standard, is there a need to define it? Please specify?	□ Yes □ No_____ □ N/A_____ □ Comments _____ _____

Please give me your email address if you wish to have a copy of the results.

I.2 Collected data from experts

The collected data for the QRs sections are listed in the following tables and the specialists who distributed the survey in their respective industry are listed.

RespondentID	CollectorID	StartDate	EndDate
1482098231	20159947	07-13-2011	07-13-2011
1474048025	20159947	07-05-2011	07-05-2011
1473558661	20159947	07-04-2011	07-05-2011
1469997474	20159947	06-29-2011	06-29-2011
1469951280	20159947	06-29-2011	06-29-2011
1468644722	20159947	06-28-2011	06-28-2011
1468630975	20159947	06-28-2011	06-28-2011

IP Address	Email Address	First Name	LastName
211.244.1.2			
210.205.122.190			
210.205.122.190			
174.94.91.54			
174.94.91.54			
174.94.91.54			
174.94.91.54			

Column1	Column2	Column3	Column4
Custom Data	**Your details**		**What is your position within your organization nowadays? Please select the best match or add an alternative.**
	Family name	**Forenames(s)**	**Response**
	AHN	sunho	
	Kim	Seong wook	Developer
	Yang	Sungname	Other, please specify
	McTeigue	Jerome	Other, please specify
	From Italy		Quality engineer
	Howard	Leanne	Quality assurance manager
	M	Taleb	Quality engineer
	Nicola Iacovelli		Quality assurance manager

Column1	Column2	Column3	Column4
What size of organisation do you work for? Please select the best match for the total number.		**For the organisation you work for, what % of the total people work on software development?**	
Response	**Comments on the total size**	**Response**	**Comments on the % working on software development**
10 - 50 people		1-9% of the total	
51 - 300 people		61-80%	
301 - 1000 people		61-80%	
>5000 people			
51 - 300 people			

Column1	Column2	Column3	Column4
What does your company do? Please specify the following activity domains, selecting answers for each row. If the choices are not suitable for your organisation please describe what your organisation does in the comments box.			
Aeronautics	Electronics	Banking	Education
Not relevant	Largest part	Largest part	Most important
		Most important	Most important

Column1	Column2	Column3	Column4
What does your company do? Please specify the following activity domains, selecting answers for each row. If the choices are not suitable for your organisation please describe what your organisation does in the comments box.			
Research and development	Health	Security	Other, please specify
Most important			
			Largest part
			Most important
	Most important		Most important
	Most important		Most important

Column1	Column2	Column3	Column4
Comments on what your company does	**What types of projects and software are developed by the company? Please indicate with are relevant and which are the most important to your company. If you do not know please state this in the comments box.**		
	Systems	Real time	Business
Certification process development			
and Quality measurement development			
	Moderately important	Not relevant	Most important
Logistics	Most important		
information & communications technology			Most important
Government, Telco			
ICT services for public administrations			
Scientific	Embedded	Personal	Internet based
Not relevant	Moderately important	Not relevant	Most important
			Most important

Column1	Column2	Column3	Column4
		What is the business critical level of your software products? (For example if the software product is related to critical systems such as fire systems and nuclear systems).	
Others, please specify	Comments on relevant and important projects	Response	Comments on business critical nature of the
			projects
	N/A		N/A
		Not critical	
		Not critical	
		Critical	
Not relevant	We test, and offer test training	Critical	
		Less critical	

Column1	Column2	Column3	Column4
Who are the stakeholders interested in processing of QRs? Please select the department that is most interested. If it is not listed Please select 'Other' and add comments on what department it is.		**Who is responsible for managing QRs for a specific software development project?**	
Response	**Comments on the stakeholders interested in processing of QRs**	**Response**	**Comments on who is responsible for specific software development projects**
Department of marketing			N/A
Department of IT or business operations		Quality engineer	
Department of IT or business operations		Other, please specify	Development manager
Others, please specify	All	Project manager	
Department of management		Project manager	Developer and Quality Engineer
Departments of management and development		Project manager and quality assurance manager	

Column1	Column2	Column3	Column4
How many years of experience does he/she have in this role?		**Please, specify the type of training given in software quality for the people involved? If you do not know, please state this in the comments box**	
Response	**Comments on who is responsible for specific software development projects**	**Project administrator**	**Project manager**
	N/A	Norms and standards	Processes and methods
1 year or more		No training	Norms and standards
1 year or more			
1 year or more			Processes and methods
Other length of experience, please specify below.	Most would have at least 2 years and Project managers and Test managers more than 5 years of experience in their respective area.		Processes and methods
1 year or more			Processes and methods

Column1	Column2	Column3	Column4
Please, specify the type of training given in software quality for the people involved? If you do not know, please state this in the comments box.			
Software or systems developer	**Quality engineer**	**Others, please specify**	**I do not know**
Processes and methods	Other training (please specify)		
Software tools	Norms and standards		
Processes and methods	Processes and methods		
Software tools	Norms and standards		

Column1	Column2	Column3	Column4
	Does your organization use a QRs process? (Identification, specification, representation, documentation and prioritization or others)? Please specify the most accurate answer.		**If your organization uses a QRs process, please specify which of the following activities are performed by your process. Please identify the most important activities. If you do not know please state this as a comment.**
Comments on software quality training	**Response**	**Comments QRs process**	Identification of quality requirements
	Use all the time		Most important
	Plan to but not yet		Not relevant
All type of training	Use all the time		Most important
	Use all the time		Most important
	Use all the time		Most important
	Use sometimes		Most important
	Use all the time		Most important

The international specialists and domain representatives who agreed to distribute the survey in their respective industry are:

Mr. Tom McBride	Australia
Ms. Alison Holt	New Zealand
Dr. Klaudia Dussa-Zieger	Germany
Dr. Jenny Dugmore	United Kingdom
Dr. Yasuharu Nishi	Japan
Prof. Keum-Suk Lee	Korea
Dr. Juan Garbajosa	Spain
Mr. Matt Mansell	New Zealand

ANNEX II

QUESTIONNAIRE ON SOQUAREM METHOD

Annex II is divided into two parts. The first one describes the survey. The second part presents collected data from the participants and software quality experts.

II.1 Description of the survey

This part describes the purpose of the survey and its detailed description

II.1.1 Purpose of the survey

Dear Sir/Madam,

To know the extent of SOQUAREM toward addressing engineering practices of quality requirements, we are conducting a survey. The survey serves to acquire indicators on the applicability of SOQUAREM process and to identify its strengths and weaknesses.

Results of the questionnaire will be useful for a further detailed evaluation step for SOQUAREM. Considering the importance of this information for our research, your contribution is very much desired for the success of our research objective. Gathered data will remain confidential and anonymous.

We thank you in advance for your collaboration which to our research.

We start the survey by a series of questions on the experience of your personnel. Later, a series of questions regrouped into 3 categories relating to:

a. Evaluating criteria of SOQUAREM process (Table1);
b. Applicability of SOQUAREM;
c. Other findings.

Table-A II-1 evaluation criteria of SOQUAREM

	Characteristics and criteria	
	Identification	**Representation and Traceability**
SOQUAREM concepts	**BMM**	**Utility tree**
	BCT	**Scenarios Template**
	Statement rules	
	Refinement rules	
	Linkage rules	
	Scenarios template	

Please answer questions by referring to your experience in the field of the quality of software product.

II.1.2 Description of the survey

This survey begins by presenting the identification form of the participants (Table- A II-2), the instructions on modalities of answers and finally describes sections of the survey as follows:

1. Identification of QAs (Table- A II-3);
2. Representation of QAs (Table- A II-4);
3. Conflict resolution of QAs (Table- A II-5);
4. Integration of QAs with FRs (Table- A II-6);
5. Applicability of SOQUAREM (Table- A II-7);
6. Other findings (Table- A II-8).

Table-A II-2 Identification of the participant

FORM Of IDENTIFICATION OF THE PARTTICIPANT	
Participant code:_____ Date (jj / mm /aaaa): _____ / _____ / _____	
Questions	**Answers**
1. What is your position nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Engineer quality ☐ Quality assurance manager ☐ Other, specify_____
2. What is your experience in the Software Requirements Engineering discipline (in years)?	☐ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☐ Other, specify_____
2. What is your experience in the Software Quality Engineering discipline (in years)?	☐ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☐ Other, specify_____

Instructions on the modalities of answers

Objective is to know if quality requirements management techniques have been addressed by SOQUAREM process and to identify its critical weaknesses as well as its improvement.
The questionnaire is organized in 3 main sections:

1. The first section deals with the evaluating criteria of SOQUAREM: identification, representation and Traceability.
2. For each criterion: evaluate Applicability, Appropriateness, Completeness and Understandability of used concepts of SOQUAREM by giving 3 question choices: 3 = very good, 2= fair, 1= poor.
3. A second section evaluates the applicability of SQOUAREM according to criteria chosen from literature.
4. A third section is dedicated to other findings where your opinions are taken into account;
5. For any remark or additional information, please write it in the commentaries section relating to asked questions;
6. It is important that your answers be based on your experience and your practices in the field of software engineering.

Sections of the survey

Table-A II-3 Identification of QAs

Identification of QAs:

Phases involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the transformation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	**Applicability** Were these concepts adequately applied to identify and derive QAs from business goals?	**Appropriateness** Were these concepts appropriately used to identify and derive QAs from business goals?	**Understandability** Were these concepts easily understandable to identify and derive QAs from business goals?	**Completeness** Were these concepts used without any missing elements that could contribute to identify QAs?
BMM				
BCT				
Scenarios template				
Statement rules Refinement rules Linkage rules				
ISO/IEC 9126				

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

Table-A II-4 Representation of QAs

Representation of QAs

Phase involved in SOQUAREM process: 4

To know to what extent SOQUAREM do address representation of quality attributes, evaluate if involving concepts like Utility tree and scenario template are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could help to represent and retrace QAs to business goals.

Concepts/criteria	**Applicability** Were these concepts adequately applied to represent and retrace QAs?	**Appropriateness** Were these concepts appropriately used to represent and retrace QAs?	**Understandability** Were these concepts easily understandable to represent and retrace QAs?	**Completeness** Were these concepts used without any missing elements that could help to represent QAs?
Utility tree				
Scenarios template				

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

Table-A II-5 Conflicts resolution of QAs

Conflicts resolution of QAs

Phase involved in SOQUAREM process: 5

To know to what extent SOQUAREM do address conflicts resolution of quality attributes, evaluate if involving concepts like Impact matrix and Weighted method are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if they are **missing elements** from involved concepts which could help to resolve conflicts among QAs.

Concepts/criteria	**Applicability** Were these concepts adequately applied to resolve conflicts among QAs??	**Appropriateness** Were these concepts appropriately used to resolve conflicts among QAs?	**Understandability** Were these concepts easily understandable to resolve conflicts among QAs?	**Completeness** Were these concepts used without any missing elements that could help to resolve conflicts among QAs?
Impact matrix				
Weighted method				

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

Table-A II-6 Integration of QAs with FRs

Integration with FRs

Phase involved in SOQUAREM process: 6

To know to what extent SOQUAREM do address integration of QAs with FRs, evaluate if involving concepts like Mapping rules are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if they are **missing elements** from involved concepts which could contribute to map QAs into the FRs process.

Concepts/criteria	**Applicability** Were Mapping rules adequately applied to integrate QAs with FRs??	**Appropriateness** Were Mapping rules appropriately used to integrate QAs with FRs?	**Understandability** Were Mapping rules easily understandable to integrate QAs with FRs?	**Completeness** Were Mapping rules used without any missing elements that could integrate QAs with FRs?
Mapping rules				

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

Table-A II-7 Applicability of SOQUAREM

Investigate applicability of SOQUAREM to quality requirements management by evaluating it according to the following criteria (Mead, 2005):

- **Adaptability to quality requirements:** the ability of SOQUAREM to manage quality requirements
- **Client acceptance:** If clients agree SOQUAREM when managing their requirements?
- **Complexity:** the degree of difficulty in understanding and properly executing SOQUAREM process. Can the requirements engineers and stakeholders easily perform SOQUAREM method correctly once they learn the process?
- **Scalability:** the ability of the SOQUAREM process to address quality requirements of enterprise-level system, in addition to smaller applications.

SOQUAREM/criteria	Adaptability to QRs	Client acceptance	Complexity	Scalability
SOQUAREM				

Scale: 3 = very good, 2= fair, 1= poor.

Table-A II-8 Other findings

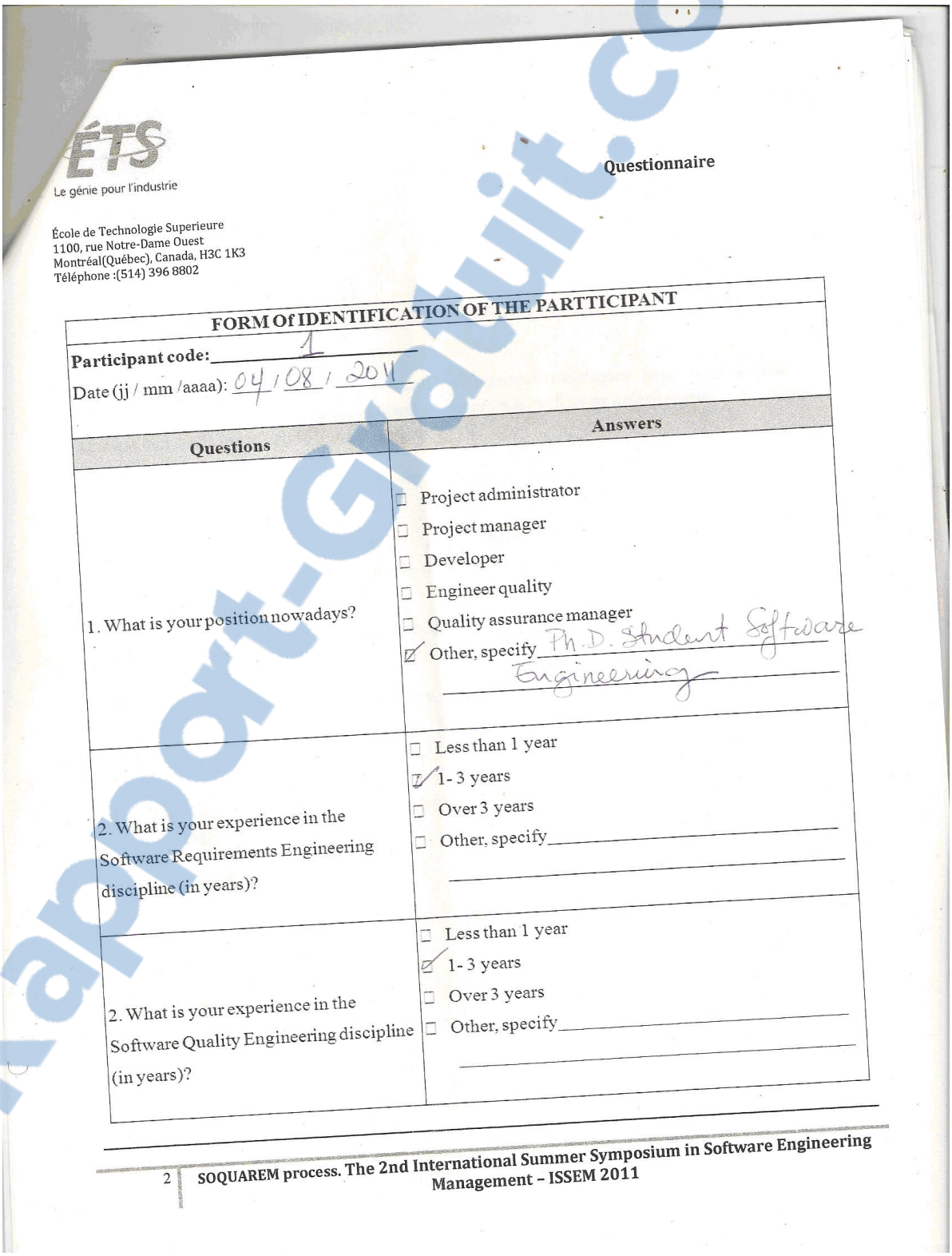
Dealing with software quality 1. To what extent does SOQUAREM process help organizations to deal with software quality? 2. Is the process easy to apply? And what is the time required to apply it? 3. How can SOQUAREM be improved? 4. What are strengths, weaknesses, opportunities and threatens?	
Other issues What challenges (i.e., cost, man-power) of applying the process would you foresee?	
Further impression Do you have any further comments?	

Thanks for your collaboration.

II.2 Collected data from participants and software quality experts

The collected responses of the participants are listed in the following tables (there are four participants with 5 response tables each). The specialists who evaluate SOQUAREM methodology are also listed (the highlighted ones) with their associated feedback.

II.2.1 Responses of the participant 1 are given in the following tables:



ÉTS
Le génie pour l'industrie

Questionnaire

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

FORM Of IDENTIFICATION OF THE PARTTICIPANT

Participant code: 1

Date (jj / mm /aaaa): 04 / 08 / 2011

Questions	Answers
1. What is your position nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Engineer quality ☐ Quality assurance manager ☑ Other, specify Ph.D. Student Software Engineering
2. What is your experience in the Software Requirements Engineering discipline (in years)?	☐ Less than 1 year ☑ 1- 3 years ☐ Over 3 years ☐ Other, specify
2. What is your experience in the Software Quality Engineering discipline (in years)?	☐ Less than 1 year ☑ 1- 3 years ☐ Over 3 years ☐ Other, specify

Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 1

Identification of QAs:

Steps involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the derivation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	Applicability Were these concepts adequately applied to identify and derive QAs from business goals?	Appropriateness Were these concepts appropriately used to identify and derive QAs from business goals?	Understandability Were these concepts easily understandable to identify and derive QAs from business goals?	Completeness Were these concepts used without any missing elements that could contribute to identify QAs?
BMM	3	3	3	2
BCT	3	2	2	2
Scenarios template	1	1	1	
Statement rules **Refinement rules** **Linkage rules**	3	3	3	3
ISO/IEC 9126	3	3	2	3

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

the 2nd Level Refinement (i.e. BG.1.1.1) is lost in the Utility Tree

Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 1

Representation of QAs
Step involved in SOQUAREM process: 3

To know to what extent SOQUAREM do address representation of quality attributes, evaluate if involving concepts like Utility tree and scenarios template are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could help to represent and retrace QAs to business goals.

Concepts/criteria	Applicability Were these concepts adequately applied to represent and retrace QAs?	Appropriateness Were these concepts appropriately used to represent and retrace QAs?	Understandability Were these concepts easily understandable to represent and retrace QAs?	Completeness Were these concepts used without any missing elements that could help to represent QAs?
Utility tree	3	2	3	2
Scenarios template	1	1	1	1

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:

The Scenario Template is Very Good (~~all~~ Applicability, Appropriateness, Understandability & Completeness) for Step 4

2. Applicability of SOQUAREM

Investigate applicability of SOQUAREM to quality requirements management by evaluating it according to the following criteria [1]:

- **Adaptability to quality requirements:** the ability of SOQUAREM to manage quality requirements
- **Client acceptance:** If clients agree SOQUAREM when managing their requirements?
- **Complexity:** the degree of difficulty in understanding and properly executing SOQUAREM process. Can the requirements engineers and stakeholders easily perform SOQUAREM method correctly once they learn the process?
- **Scalability:** the ability of the SOQUAREM process to address quality requirements of enterprise-level system, in addition to smaller applications.

SOQUAREM/criteria	Adaptability to QRs	Client acceptance	Complexity	Scalability
SOQUAREM	3	3	3	3

Scale: 3 = very good, 2= fair, 1= poor.

1. Mead, N. R (2005) "Security Quality Requirements Engineering (SQUARE) Methodology." Technical report CMU/SEI-2005-TR-009. ESC-TR-2005-009.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 1

3. Other findings	
Dealing with software quality 1. To what extent does SOQUAREM process help organizations to deal with software quality? 2. Is the process easy to apply? And what is the time required to apply it? 3. How can SOQUAREM be improved? 4. What are strengths, weaknesses, opportunities and threatens?	It allows to represent in a very structured and simple to use way all the relevant concepts, and to trace backwards to the high-level needs that caused each Q·A. ⇒ Developing a tool ⇒ Measure the benefits in real cases.
Other issues What challenges (i.e., cost, man-power) of applying the process would you foresee?	⇒ It requires an additional effort from the Requirements professional, so, some companies might not be willing to finance the costs involved
Further impression Do you have any further comments?	

Thanks for your collaboration.

The study is anonymous and data collected does not contain any identifying information.
If you have questions about this study you may contact
Rachida.djouab.1@ens.etsmtl.ca

If you have other questions, you can contact the ETS Ethics Committee for Research (CÉR) that examines and approves the modalities of studies such as this one. The chair of the CÉR can be reached at : +1 (514) 396-8829.

Si vous avez d'autres questions, vous pouvez contacter le Comité d'éthique de la recherche (CÉR) de l'ÉTS qui examine et approuve les modalités des études comme celle-ci. Pour rejoindre le président du CÉR, composez le (514) 396-8829.

II.2.2 Responses of the participant 2 are given in the following tables:

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: _____

Identification of QAs:
Steps involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the derivation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	Applicability Were these concepts adequately applied to identify and derive QAs from business goals?	Appropriateness Were these concepts appropriately used to identify and derive QAs from business goals?	Understandability Were these concepts easily understandable to identify and derive QAs from business goals?	Completeness Were these concepts used without any missing elements that could contribute to identify QAs?
BMM	3	3	3	3
BCT	3	3	3	3
Scenarios template	2	2	1	1
Statement rules **Refinement rules** **Linkage rules**	3 3 2	3 3 2	3 3 2	3 3 2
ISO/IEC 9126	1	1	1	1

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:
According to my perspective, The scenarios templates should provide more information to guide new users of SO-QUAREM for me it is not clear how you select or map the quality attribute from ISO/IEC 9126

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

FORM Of IDENTIFICATION OF THE PARTTICIPANT	
Participant code:_____ Date (jj / mm /aaaa): 19/08/2011	
Questions	**Answers**
1. What is your position nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Engineer quality ☐ Quality assurance manager ☒ Other, specify Phd Student
2. What is your experience in the Software Requirements Engineering discipline (in years)?	☐ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☒ Other, specify only in Teaching Software Engineering for undergrads
2. What is your experience in the Software Quality Engineering discipline (in years)?	☐ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☒ Other, specify Teaching software engineering for undergrads

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: _____

Representation of QAs
Step involved in SOQUAREM process: 3

To know to what extent SOQUAREM do address representation of quality attributes, evaluate if involving concepts like Utility tree and scenarios template are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could help to represent and retrace QAs to business goals.

Concepts/criteria	Applicability Were these concepts adequately applied to represent and retrace QAs?	Appropriateness Were these concepts appropriately used to represent and retrace QAs?	Understandability Were these concepts easily understandable to represent and retrace QAs?	Completeness Were these concepts used without any missing elements that could help to represent QAs?
Utility tree	3	3	3	3
Scenarios template	2	2	1	1

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:
Scenarios templates : The same comments written in pg. 5

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: _____

2. Applicability of SOQUAREM

Investigate applicability of SOQUAREM to quality requirements management by evaluating it according to the following criteria [1]:

- **Adaptability to quality requirements:** the ability of SOQUAREM to manage quality requirements
- **Client acceptance:** If clients agree SOQUAREM when managing their requirements?
- **Complexity:** the degree of difficulty in understanding and properly executing SOQUAREM process. Can the requirements engineers and stakeholders easily perform SOQUAREM method correctly once they learn the process?
- **Scalability:** the ability of the SOQUAREM process to address quality requirements of enterprise-level system, in addition to smaller applications.

SOQUAREM/criteria	Adaptability to QRs	Client acceptance	Complexity	Scalability
SOQUAREM	3	1/2	2	2

I think that it can be used for both but if the application is small maybe people will not want to use it

Scale: 3 = very good, 2= fair, 1= poor.

clients never have time and the use of SOQUAREM will demand time

it is simple to understand but requires time

1. Mead, N. R (2005) "Security Quality Requirements Engineering (SQUARE) Methodology." Technical report CMU/SEI-2005-TR-009. ESC-TR-2005-009.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: _____

3. Other findings	
Dealing with software quality 1. To what extent does SOQUAREM process help organizations to deal with software quality? 2. Is the process easy to apply? And what is the time required to apply it? 3. How can SOQUAREM be improved? 4. What are strengths, weaknesses, opportunities and threatens?	It helps to get to the point & discover the requirements. It is easy to understand but it takes time. It can take a lot of time because managers don't have time to define goals clearly. Maybe they know what they want but it is not written anywhere.
Other issues What challenges (i.e., cost, man-power) of applying the process would you foresee?	→ To develop an automated tool → time + cost for gathering requirements
Further impression Do you have any further comments?	

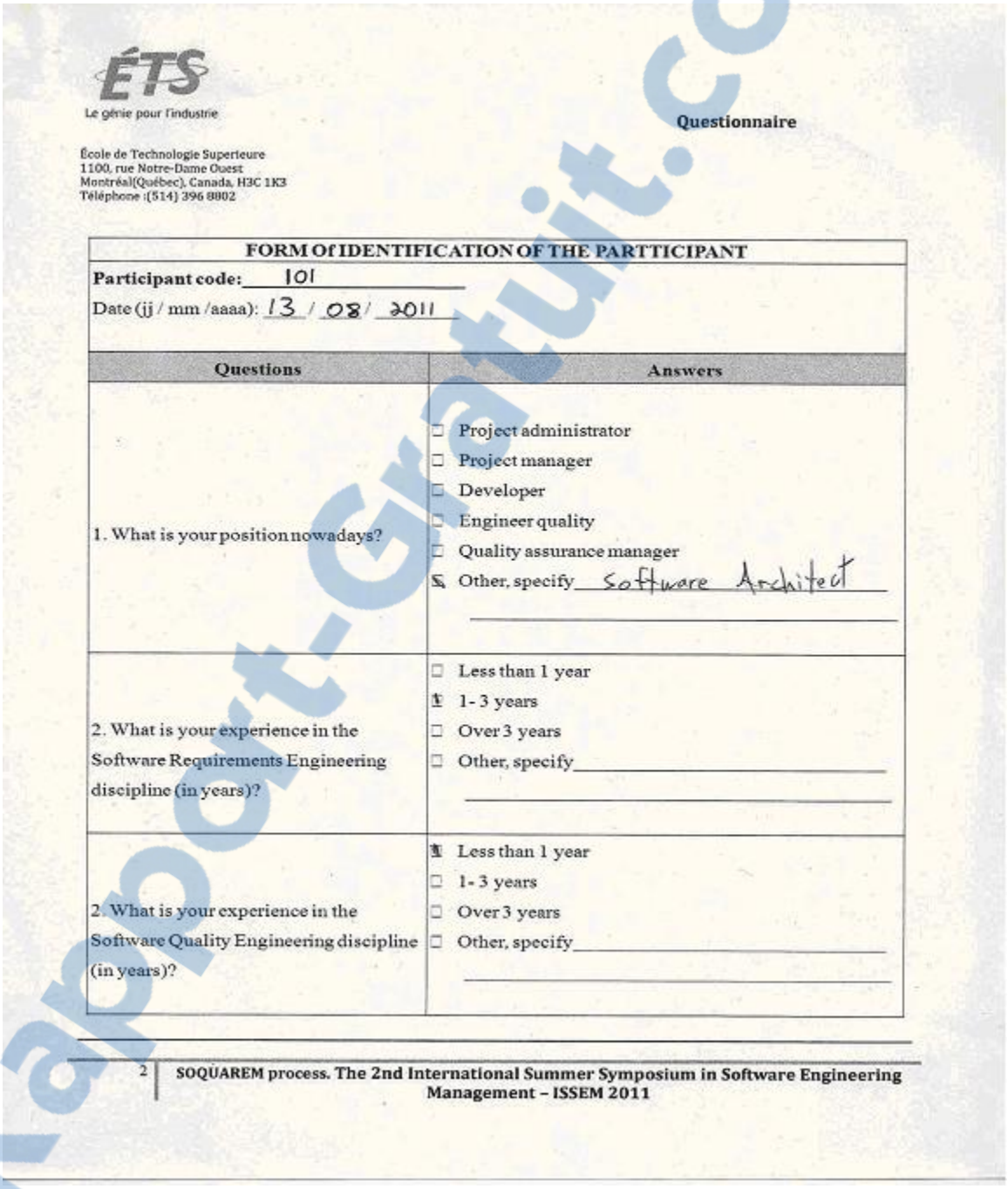
Thanks for your collaboration.

The study is anonymous and data collected does not contain any identifying information.
If you have questions about this study you may contact
Rachida.djouab.1@ens.etsmtl.ca

If you have other questions, you can contact the ETS Ethics Committee for Research (CÉR) that examines and approves the modalities of studies such as this one. The chair of the CÉR can be reached at : +1 (514) 396-8829.

Si vous avez d'autres questions, vous pouvez contacter le Comité d'éthique de la recherche (CÉR) de l'ÉTS qui examine et approuve les modalités des études comme celle-ci. Pour rejoindre le président du CÉR, composez le (514) 396-8829.

II.2.3 Responses of the participant 3 are given in the following tables:



ÉTS
Le génie pour l'industrie

Questionnaire

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

FORM Of IDENTIFICATION OF THE PARTTICIPANT

Participant code: 101

Date (jj / mm /aaaa): 13 / 08 / 2011

Questions	Answers
1. What is your position nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Engineer quality ☐ Quality assurance manager ☒ Other, specify Software Architect
2. What is your experience in the Software Requirements Engineering discipline (in years)?	☐ Less than 1 year ☒ 1- 3 years ☐ Over 3 years ☐ Other, specify
2. What is your experience in the Software Quality Engineering discipline (in years)?	☒ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☐ Other, specify

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 101

Identification of QAs:
Steps involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the derivation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	Applicability Were these concepts adequately applied to identify and derive QAs from business goals?	Appropriateness Were these concepts appropriately used to identify and derive QAs from business goals?	Understandability Were these concepts easily understandable to identify and derive QAs from business goals?	Completeness Were these concepts used without any missing elements that could contribute to identify QAs?
BMM	2	2	2	2
BCT	2	2	2	2
Scenarios template	~~1~~ 3	3	2	3
Statement rules **Refinement rules** **Linkage rules**	3	3	2	3
ISO/IEC 9126	1	2	1	2

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:

I don't see an appropriate way to deal with constraint

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 10

Identification of QAs:
Steps involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the derivation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	Applicability Were these concepts adequately applied to identify and derive QAs from business goals?	Appropriateness Were these concepts appropriately used to identify and derive QAs from business goals?	Understandability Were these concepts easily understandable to identify and derive QAs from business goals?	Completeness Were these concepts used without any missing elements that could contribute to identify QAs?
BMM	2	2	2	2
BCT	2	2	2	2
Scenarios template	3	3	2	3
Statement rules Refinement rules Linkage rules	3	3	2	3
ISO/IEC 9126	1	2	1	2

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:

I don't see an appropriate way to deal with constraint

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 101

2. Applicability of SOQUAREM

Investigate applicability of SOQUAREM to quality requirements management by evaluating it according to the following criteria [1]:

- **Adaptability to quality requirements:** the ability of SOQUAREM to manage quality requirements
- **Client acceptance:** If clients agree SOQUAREM when managing their requirements?
- **Complexity:** the degree of difficulty in understanding and properly executing SOQUAREM process. Can the requirements engineers and stakeholders easily perform SOQUAREM method correctly once they learn the process?
- **Scalability:** the ability of the SOQUAREM process to address quality requirements of enterprise-level system, in addition to smaller applications.

Figure 3 is very good

SOQUAREM/criteria	Adaptability to QRs	Client acceptance	Complexity	Scalability
SOQUAREM	3	1	2	3

but find a table to summary refined business goal

Scale: 3 = very good, 2= fair, 1= poor.

We have to tell the customers why Quality Attribute are important

Figure 1 have too much details

a more simple figure can be add (same for Figure)

1. Mead, N. R (2005) "Security Quality Requirements Engineering (SQUARE) Methodology." Technical report CMU/SEI-2005-TR-009. ESC-TR-2005-009.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: 101

3. Other findings	
Dealing with software quality 1. To what extent does SOQUAREM process help organizations to deal with software quality? 2. Is the process easy to apply? And what is the time required to apply it? 3. How can SOQUAREM be improved? 4. What are strengths, weaknesses, opportunities and threatens?	1. Requirements Management & Change request to Help to priorize development effort 2. Yes. 1 hour. 3. Do Marketing to present advantage of Quality Attribut requirement management 4. threatens : How this link to Agile Method? Strengths : The utility tree
Other issues What challenges (i.e., cost, man-power) of applying the process would you foresee?	- Marketing of the method - Link the Method to QA Model & Measurement
Further impression Do you have any further comments?	Very Very good high Quality Document! Thank you very much for this sharing of knowledge !! continue your good work

This adress should be more visible in the document

Thanks for your collaboration.

The study is anonymous and data collected does not contain any identifying information.
If you have questions about this study you may contact
Rachida.djouab.1@ens.etsmtl.ca

If you have other questions, you can contact the ETS Ethics Committee for Research (CÉR) that examines and approves the modalities of studies such as this one. The chair of the CÉR can be reached at : +1 (514) 396-8829.

Si vous avez d'autres questions, vous pouvez contacter le Comité d'éthique de la recherche (CÉR) de l'ÉTS qui examine et approuve les modalités des études comme celle-ci. Pour rejoindre le président du CÉR, composez le (514) 396-8829.

II.2.4 Responses of the participant 4 are given in the following tables:

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

FORM Of IDENTIFICATION OF THE PARTTICIPANT	
Participant code: K Date (jj / mm /aaaa): 04 / 08 / 2011	
Questions	**Answers**
1. What is your position nowadays?	☐ Project administrator ☐ Project manager ☐ Developer ☐ Engineer quality ☐ Quality assurance manager ☑ Other, specify PhD
2. What is your experience in the Software Requirements Engineering discipline (in years)?	☐ Less than 1 year ☐ 1- 3 years ☑ Over 3 years ☐ Other, specify
2. What is your experience in the Software Quality Engineering discipline (in years)?	☑ Less than 1 year ☐ 1- 3 years ☐ Over 3 years ☐ Other, specify

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: K

Identification of QAs:

Steps involved in SOQUAREM process: 1, 2 and 3

To know to what extent SOQUAREM do address identification of quality attributes, evaluate if involving concepts like BMM, BCT and the derivation rules (Statement, Refinement and Linkage rules) are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could contribute to identify QAs?

Concepts/criteria	Applicability Were these concepts adequately applied to identify and derive QAs from business goals?	Appropriateness Were these concepts appropriately used to identify and derive QAs from business goals?	Understandability Were these concepts easily understandable to identify and derive QAs from business goals?	Completeness Were these concepts used without any missing elements that could contribute to identify QAs?
BMM	3	1	3	2
BCT	1	3	3	3
Scenarios template	2	2	2	3
Statement rules Refinement rules Linkage rules	2	1	2	2
ISO/IEC 9126	3	3	2	3

Scale: 3 = very good, 2= fair, 1= poor.

Commentaries:

N/A.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: K

Representation of QAs
Step involved in SOQUAREM process: 3

To know to what extent SOQUAREM do address representation of quality attributes, evaluate if involving concepts like Utility tree and scenarios template are **adequately applied** and **appropriately used** in SOQUAREM case and **easily understandable**. Evaluate also if there are **missing elements** from involved concepts which could help to represent and retrace QAs to business goals.

Concepts/criteria	Applicability Were these concepts adequately applied to represent and retrace QAs?	Appropriateness Were these concepts appropriately used to represent and retrace QAs?	Understandability Were these concepts easily understandable to represent and retrace QAs?	Completeness Were these concepts used without any missing elements that could help to represent QAs?
Utility tree	3	3	3	3
Scenarios template	2	3	3	2

Scale: 3 = very good, 2= fair, 1= poor.
Commentaries:

N/A.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: K

2. Applicability of SOQUAREM

Investigate applicability of SOQUAREM to quality requirements management by evaluating it according to the following criteria [1]:

- **Adaptability to quality requirements:** the ability of SOQUAREM to manage quality requirements
- **Client acceptance:** If clients agree SOQUAREM when managing their requirements?
- **Complexity:** the degree of difficulty in understanding and properly executing SOQUAREM process. Can the requirements engineers and stakeholders easily perform SOQUAREM method correctly once they learn the process?
- **Scalability:** the ability of the SOQUAREM process to address quality requirements of enterprise-level system, in addition to smaller applications.

SOQUAREM/criteria	Adaptability to QRs	Client acceptance	Complexity	Scalability
SOQUAREM	3	3	2	3

Scale: 3 = very good, 2= fair, 1= poor.

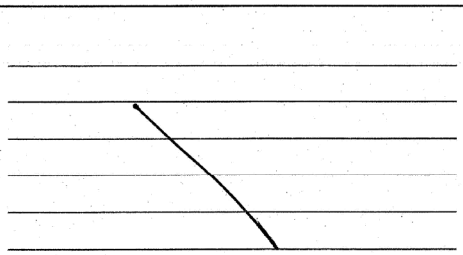
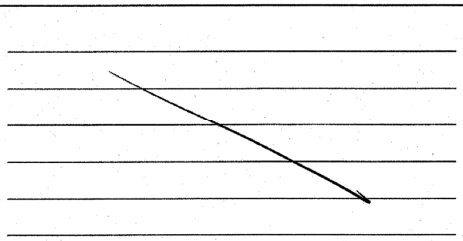
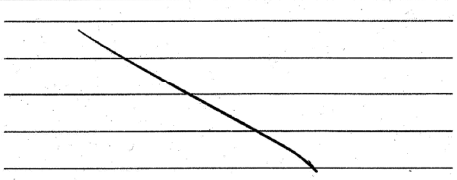
1. Mead, N. R (2005) "Security Quality Requirements Engineering (SQUARE) Methodology." Technical report CMU/SEI-2005-TR-009. ESC-TR-2005-009.

ÉTS
Le génie pour l'industrie

École de Technologie Superieure
1100, rue Notre-Dame Ouest
Montréal(Québec), Canada, H3C 1K3
Téléphone :(514) 396 8802

Questionnaire

Participant code: K

3. Other findings	
Dealing with software quality 1. To what extent does SOQUAREM process help organizations to deal with software quality? 2. Is the process easy to apply? And what is the time required to apply it? 3. How can SOQUAREM be improved? 4. What are strengths, weaknesses, opportunities and threatens?	
Other issues What challenges (i.e., cost, man-power) of applying the process would you foresee?	
Further impression Do you have any further comments?	

Thanks for your collaboration.

The study is anonymous and data collected does not contain any identifying information.
If you have questions about this study you may contact
Rachida.djouab.1@ens.etsmtl.ca

If you have other questions, you can contact the ETS Ethics Committee for Research (CÉR) that examines and approves the modalities of studies such as this one. The chair of the CÉR can be reached at : +1 (514) 396-8829.

Si vous avez d'autres questions, vous pouvez contacter le Comité d'éthique de la recherche (CÉR) de l'ÉTS qui examine et approuve les modalités des études comme celle-ci. Pour rejoindre le président du CÉR, composez le (514) 396-8829.

The international specialists and scientist who agreed to evaluate SOQUAREM methodology are:

Dr. Tafline Murnane	Australia
Ms. Alison Holt	New Zealand
Dr. Klaudia Dussa-Zieger	Germany
Dr. Annette Reilly	USA
Dr. Jenny Dugmore	UK
Mr. Anatol Kark	Canada
Prof. Motoei Azuma	Japan
Prof. Keum-Suk Lee	Korea
Dr. Nigel Bevan	UK
Dr. Juan Garbajosa	Spain

The highlighted ones have given their feedback on SOQUAREM in the following sections:

1. Feedback of Dr. Annette Reilly USA is:

From: "Reilly, Annette D" <annette.d.reilly@lmco.com>
Date: July 20, 2011 8:48:50 AM EDT
To: witold suryn <witold.suryn@etsmtl.ca>
Subject: RE: EXTERNAL: SOQUAREM methodology evaluation

Hello Witold,
Perhaps this is not too late to be helpful.
Overall the presentation is well organized, thoroughly detailed in the example, and demonstrates a sound method.

All the references to 9126 should be checked to determine if there is a current reference in the ISO/IEC 25000 SQARE series to replace them.

In a few places, the text is worded as if the method performed itself, minimizing the amount of judgment needed to produce the results.
This is a concern in the weighting of quality attributes (p.9, Table 5.5, and section 1.2.2.7, and p. 63.) since weighting is either done before the evaluation, when stakeholders have incomplete information, and/or revised afterwards, when the weighting is used to reverse engineer the preferred trade-off. Identifying how to spread equitably costs and benefits for all the stakeholders is hard to subject to a quality process. Also, "conflicts among quality attributes are resolved by attributing weights to the cells of the matrix". This does not necessarily produce the result.
It would be good to discuss the use of automated modeling systems for decision support in this regard.

I would suggest having an English-language copy editor check the text, especially for that most idiosyncratic of problems, use of the article in English.

Thanks for the offer; I appreciated the opportunity to review this work.

The opinions in this message do not represent an official view of my employer or of the US TAG.

Annette Reilly
Lockheed Martin Information Systems & Global Solutions
Business Development
301.337-4409
Cell 240.888.4348
annette.d.reilly@lmco.com

2. Feedback of Prof. Motoei Azuma (Japan) is:

On 2011-06-18, at 5:42 AM, Motoei_AZUMA wrote:

Dear Witold,

I did quick review on the paper.

The first impression is that the paper is well structured and easy to read.

Before going into more detailed review, I have a couple of questions,
a comment, and an information that may of your interest.

1) Q1: What is the main purpose of this paper? Thesis? Or a part of a book?

2) Q2: The paper started from Chapter 5, and end by Chapter 6.
How are the other Chapters? Shall I supposed to review only two Chapters
without knowing any on the other Chapters?

3) C1: "Business goal" looks like one of the most important concept.
However, there is no rigorous definition of it, nor good examples.
Though Chapter 6 is a case study, there is no description about the
business goal of the case.
The only part that explain the "business goal" is Page 21, Step 2.

Step 2: Refine business goals: business goals are detailed according to additional business
information such as organizational culture, regulations and guidelines, technological
constraints and business strategies to achieve business goals.

4) Info: I was asked as a program committee member and Key Note Speaker
of WoSQ 2011, which will be held in Szeged, Hungary on September 4th.

http://sites.google.com/site/wosq2011/cfp

Though the deadline date was over, I received that the committee decided to
extend the dead line by June 22.

As I think the paper meet the purpose of the workshop, will you think about
to make the paper in short and submit it?

Regards,
==========
Motoei AZUMA
Convener, ISO/IEC JTC1/SC7/WG6
ISO/IEC 250nn SQuaRE series prime project editor
Emeritus Professor, Waseda University
E-Mail: <az-mo@mtd.biglobe.ne.jp>
==========

3. Feedback of Prof. Keum-Suk Lee (Korea) is:

Comments on SOQUAREM

1. The overall steps of SOQUAREM are reasonable and applicable for the derivation of quality requirements. Especially the BMM and BCT, consensus session and Quality attribute list could improve identification and derivation of quality attributes (characteristics) from business goals.
2. But the template for specifying quality attributes(table 5.6, table 6.19) should be revised in the following points;
   A. The 'quality attribute name' field is missing in table 5.6.
   B. Not only developer but also quality evaluator is one of the target stakeholders of relevant QA, so this template should be useful for both of them.
   C. The 'Representation' item should be refined to be more understandable from software engineer's viewpoints. For example the column title of 'list of sub quality attributes' is not consistent with its contents(refined business goals), and it would be convenient for the evaluator to represent 'the priority of each scenario' more specifically as values(including the above 'priority' item).
   D. 'Activities and phases' item could be described more precisely considering the software lifecycle concept or software process standards.
3. In my experience, the resolving of conflicting attributes is very important and difficult to implement. In fact, these QA priorities will affect the cost and efforts of software development and evaluation processes.
4. Without reviewing the other chapters, I am afraid to comment the details.
   Good luck!

BIBLIOGRAPHY

Abran, A., Moore, J.W., Bourque, P., Dupuis, R., Tripp, L. 2004. “Guide to the Software Engineering Body of Knowledge”. IEEE Computer Society/ ISO/IEC JTC1 SC7. Latest version downloadable from www.swebok.org

Aburub F., M. Odeh and I. Beeson. 2007 «Modelling non-functional requirements of business processes **",** Journal Information and Software Technology., Vol. 49, (Butterworth-Heinemann Newton, MA, USA November. 2007), p.1162–1171.

Araujo J., A. Moreira, I. Brito, and A. Rashid. 2002 "Aspect-Oriented Requirements with UML," presented at Workshop on "Aspect-oriented modeling with UML", UML 2002. (Dresden, Germany).

Araújo J., P.Coutinho 2003. "Identifying Aspectual Use Cases Using a Viewpoint-Oriented Requirements Method". Early Aspects 2003. Aspect-Oriented Requirements Engineering and Architecture Design, Workshop of the 2nd International Conference on Aspect-Oriented Software Development. (Boston, USA, 17 March. 2003), p. 1-6.

Arnon R. 2006. "ATAM: Introduction to SEI’s Architecture Tradeoff Analysis Method". In http://arnon.me/presentations-papers-articles/. Consulted in 2009.

Azuma M. 2001 "SQuaRE -Software Product Quality Requirements and Evaluation-". Revision of WG6 N474 based on discussion at the SC7/WG6 Prague Meeting ISO/IEC JTC1/SC7/WG6.

Azuma M. 2004 "Applying ISO/IEC 9126-1 quality model to quality requirements engineering on critical software". Sixth International Workshop on Requirements for High Assurance Systems (RHAS 2004). (Kyoto, Japan, 2004), p. 3-10.

Bart Venckeleer. 2006. ATAM "Architecture Trade-off Analysis Method" with case study. http://www.docstoc.com/docs/22734205/ATAM-Architecture-Trade-off-Analysis-Method-with-case-study. Consulted in January 2010.

Barbacci 2003. "Software-Quality Attributes and Architecture Trade-Offs.".Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

Bass L, Clements P, Kazman R. 2003. "Software Architecture in Practice ", 2nd edition, Addison-Wesley. 528 p.

Beck K. 1994. "Patterns and software development" *Dr. Dobb’s Journal*, vol. 19, no 2, p.18–23.

Bevan N., A. Motoei. 1997. "Quality in use: Incorporating human factors into the software engineering lifecycle" in Proceedings of the IEEE International Software Engineering Standards Symposium (ISESS 97). (Walnut Creek, CA June 01- 06 1997), p 169-179.

Bevan N. 1999 "Quality in use: Meeting user needs for quality". Journal of Systems and Software , vol.49, n° 3, p. 89-96.

Birk, R. van Solingen, J. Järvinen. 1998 "Business Impact, Benefit, and Cost of Applying GQM in Industry" An In-Depth, Long-Term Investigation at Schlumberger RPS. Software Metrics Symposium, 1998. Metrics 1998. Proceedings. Fifth International, (IEEE METRICS 1998). (Bethesda, MD, USA. 20-21 Nov 1998), p. 93-96.

Boehm, B., J. Brown, H. Kaspar, M. Lipow, G. MacLeod and M. Merritt. 1978 "*Characteristics of Software Quality"*, TRWSeries of Software Technology, TRWSystems and Energy, Inc. (1973); also published by North-Holland, Amsterdam.

Boehm B., H. In, T.Rodgers and Deutsch M. 2001 "Applying WinWin to quality requirements: a case study " in Proceedings of the 23rd International Conference on Software Engineering, IEEE Computer Society. (Toronto, Ontario, Canada), p. 555-564.

Borland, the OPEN ALM COMPANY. 2007. "Driving Quality Throughout the Software Delivery Lifecycle. The benefits of Lifecycle Quality Management." White Paper. www.borland.com/resources/en/pdf/solutions/lqm_driving_quality.pdf. Retrieved in 2012.

Boucké N, Weyns.D, Schelfthout.K, Holvoet.T. 2006. "Applying the ATAM to an Architecture for Decentralized Control of a Transportation System". QoSA 2006, p.180-198.

Bredemeyer D. and R. Malan. 2001 "Defining Non-Functional Requirements», in Architecture Resources for Enterprise Advantage. B. Consulting, Ed.: Bredemeyer Consulting. http://www.bredemeyer.com. Consulted in 2009.

Brito I., Moreira A. and Araujo J. 2002. "A Requirements Model for Quality Attributes," presented at Workshop on "Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design", 1st International Conference on Aspect-Oriented Software Development. (University of Twente, Enschede, Holland. April 22-26), p. 1-6.

Brito I. and Moreira A. 2003. "Towards a Composition Process for Aspect-Oriented Requirements," presented at Workshop of the 2nd International Conference on Aspect-Oriented Software Development. (Boston, Massachusetts), p. 1-6.

Business Rules Group (BRG). 2007. "Business Motivation Model". (Version 1.3). Retrieved 2010 from http://www.businessrulesGroup.org.

Carvallo J.P., X. Franch 2002a. "A Quality-Model-Base approach for describing and evaluating software packages". In Proceeding RE '02 Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering. (Essen, Germany 9-13 September), p.104-111.

Carvallo J.P., P. Botella, X.B.Illa, X.Franch, C.Quer 2002b. "Using Quality Models for Assessing COTS Selection". (WER 2002), p. 263-277.

Carvallo J.P, X. Franch. 2003. "Using Quality Models in Software Package Selection", IEEE Software. Vol. 20, no 1, p.34-41.

Carvallo J.P., X. Franch, G. Grau, C. Quer. 2004. "QM: A Tool for Building Software Quality Models", RE 2004, p.358-359.

Chung L., B. A. Nixon and E. Yu. 1994. "Using Quality Requirements to Systematically Develop Quality Software,". Proc., 4th International Conference on Software Quality. (McLean, VA, U.S.A. Oct). p. 3-5.

Chung L. and B. A. Nixon. 1995. "Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach,"; Proc., IEEE 17th International Conference on Software Engineering. (Seattle, April 24-28), p. 25-37.

Chung L, B. Nixon, E. Yu, and J. Mylopoulos. 2000. Non-Functional Requirements in Software Engineering. Kluwer Academic Publishing, Boston Hardbound, 472 p.

Constantine L. 1997 "The case for Essential Use Cases" Object Magazine. SIGS Publications NY. Vol. 7, no 3, p. 72-70.

Cooper Kendra, Lirong Dai, Yi Deng. 2004. "Performance Modeling and Analysis of Software Architectures: An Aspect-Oriented UML Based Approach". Software Engineering Research and Practice. p. 111-120.

Crosby Phil. 1979. Quality is Free. New York: McGraw-Hill. ISBN 0-07-014512-1.

Cysneiros L.M, J.C. Sampaio do Prado Leite. 2004. “Non functional Requirements: From Elicitation to Conceptual Models” IEEE Transactions on Software Engineering, vol. 30, No. 5. P.328-349. Published by the IEEE Computer Society 2004.

Dai, L., Cooper, K. 2003. “Process Definition for the Formal Design Analysis Framework Creating an Aspect-oriented Design Supporting Response Time Performance". Technical Report UTDCS-20-03. Department of Computer Science Univ. of Texas at Dallas Richardson, 37 p.

Dai L. 2005. “Formal design analysis framework: an aspect-oriented architectural framework”. PhD Dissertation, The University of Texas at Dallas, 318 p.

Dai L., Cooper K. 2005. “Modeling and Analysis of Non-functional Requirements as Aspects in a UML Based Architecture Design”. Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN’05). (Towson university, Towson, Maryland, USA May 23-25), p.178-183.

Dai Lirong, Kendra Cooper. 2006. "Helping to Meet the Security Needs of Enterprises: Using FDAF to Build RBAC into Software Architectures". Software Engineering Research and Practice 2006: 790-796.

Deng X., 2006. "Intentional Modeling for Enterprise Architecture – Managing Knowledge about “Why” to Support Change". Master’s Thesis, Faculty of Information Studies, University of Toronto. 171 p.

Dieter Landes. 1998. "Requirements Engineering for Quality Requirements – Industrial Problem Statement", Fourth International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ’98. (Pisa, Italy), p.185-186.

Djouab R., Suryn W. 2006. “An ISO/IEC standards-based quality requirements definition approach: Applicative analysis of three quality requirements definition methods” was published to ISIE 2006 Annual Conference of the IEEE Industrial Electronics Society. (9-13 July Montreal), p.3231-3239.

Djouab R., Suryn W. 2007a. “Analysis of a probabilistic quality method for evaluation of non functional requirements” was published to ICSSEA International Conference on Software and Systems Engineering and their Applications. 4-6 December - Conservatoire National des Arts et Métiers - Paris, France.

Djouab R., Suryn W. 2007b. “Applicability analysis of two quality requirements treatment methods: IESE NFR and FDAF” was published to ICSSEA International Conference on Software and Systems Engineering and their Applications. 4-6 December 2007 -Conservatoire National des Arts et Métiers - Paris, France

Djouab R., Suryn W. 2011a. "SOQUAREM: SOftware QUAlity Requirements Engineering Method". SQM Conference on Quality Management. (18-20 April 2011. Loughborough University, Leicestershire, UK).

Djouab R., Suryn W. 2011b. "Applicability of SOQUAREM method: an illustrative case study". SQM Conference on Quality Management. (18-20 April 2011. Loughborough University, Leicestershire, UK).

Doerr J., K erkow D., B. Paech. 2003. "Eliciting Efficiency Requirements with Use Cases", 9th International Workshop on Requirements Engineering. Foundation for Software Quality. (Workshop held at CaiSE' 03, June), p.23-32.

Doerr J., D. Kerkow, T. Koenig, T. Olsson, and T. Suzuki. 2005. "Non-Functional Requirements in Industry - Three Case Studies Adopting an Experience-based NFR Method". In Proceedings 13th IEEE International Conference on Requirements Engineering. (Paris France 29 August-2 September), p. 373–384.

Doerr J. 2011." Non-functional Requirements". Lecture: Requirements Engineering WS 2010/2011. Retrieved 23 november 2011 from http://wwwagse.informatik.uni-kl.de/teaching/re/ws2010/Vorlesung%20RE_WS1011_NFR.pdf. 29 p.

Doi K. 1999. “An extraction method of quality requirements in the offline requirements capturing method”, in *Transactions of the Information Processing Society of Japan.* Vol. 40, n° 11, p. 4012-20.

Dromey R.Geoff. 1995. "A Model for Software Product Quality". In Journal of IEEE Transactions on Software Engineering. (EEE Press Piscataway, NJ, USA), Vol. 21, n° 2, p. 146-162.

Egyed A. and P. Grunbacher. 2004. "Identifying requirements conflicts and cooperation: how quality attributes and automated traceability can help". In IEEE Computer Society. Vol. 21, n° 6, p.50-58.

Empress. 2004. http://www.empress-itea.org/deliverables/D3.5_v1.0_Public_Version.pdf. Retrieved in 2008.

Felici M., A. Pasquini, and S. De Panfilis. 1998. "Software Quality in User-Centred Design". ESCOM-ENCRESS 98, p. 239-247.

Felici M., M.-A. Sujan, and M. Wimmer. 2000. "Integration of Functional Cognitive and Quality Requirements: A Railways Case Study". ESCOM - SCOPE 2000, p. 395-403.

Fenton N., P. Krause and M. Neil 2001. "A probabilistic model for software defect prediction », unpublished manuscript available from the authors.

Fenton N. E., Krause, P., and Neil, M. 2002. Probabilistic Modelling for Software Quality Control. *Journal of Applied Non-Classical Logics.* Vol. 12, n° 2, p. 173-188.

Firesmith D. 2003. “Using Quality Models to Engineer Quality Requirements”. Journal of Object Technology (JOT). Vol. 2. no5 (September-October 2003), p.67-75. http://www.jot.fm/issues/issue_2003_09/column6. Retrieved in 2007.

Firesmith, Donald; Mead, Nancy R.; and Woody, Carol. 2004. "System Quality Requirements Engineering (SQUARE) Project". http://www.cert.org/sse/square.html. Retrieved in 2007.

Firesmith D. 2005. “Quality Requirements Checklist”, in Journal of Object Technology. Vol. 4, no. 9 (November-December 2005), p. 31-38, http://www.jot.fm/issues/issue_2005_11/column4. Retrieved in 2007.

Gallagher B.P 2000. "Using the architecture tradeoff analysis method to evaluate reference architecture: A case study". Technical Report CMU/SEI-2000-TN-007, Carnegie Mellon University, Software Engineering Institute. 25 p.

Herrmann, A. Barbara Paech 2007a. "MOQARE: misuse-oriented quality requirements engineering" in Requirements Engineering Journal, Vol. 13, no 1, p. 73-86.

Herrmann, A. Kerkow, D. Doerr, J. 2007b. "Exploring the Characteristics of NFR Methods" - A Dialogue About Two Approaches. , REFSQ, LNCS 4542. (Springer Verlag Berlin Heidelberg), p. 320-334.

Hill R., J. Wang, and K. Nahrstedt 2004. "Quantifying non-functional requirements: a process oriented approach," presented at Requirements Engineering Conference Proceedings. 12th IEEE International. (6-11 Sept), p. 352 - 353.

Humphrey W. 1989. Managing the software process. Addison-Wesley. 512 p.

Humphrey W. 1995. A Discipline for Software Engineering. Addison-Wesley Publishing Company, Massachusetts. 789 p.

ISO/IEC 9126. 1999-2004. - Software Engineering - Product quality. Parts 1-4.

ISO/IEC 14598. 1999. Information Technology—Software Product Evaluation, Parts 1–5. Genève, ISO/IEC.

ISO/IEC 25030 2007. - Software engineering – Software quality requirements and evaluation (SQuaRE) – Quality requirements.

ISO/IEC 15288 2002- Information Technology - Life Cycle Management - System Life Cycle Processes.

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

Jacobs S. 1999. "Introducing Measurable Quality Requirements: A Case Study", Ericsson Eurolab Deutschland. Fourth IEEE International Symposium on Requirements Engineering (RE'99). (Limerick, Ireland June 07- 11), 172 p.

Jacobson, I., et al. 1992. Object-oriented Software Engineering: A Use Case Driven Approach. Addison Wesley. Reading, MA. 582 p.

Jacobson I., Booch G., and Rumbaugh J. 1998. "Unified Modeling Language 1.3", White paper, Rational Software Corp.

Jacobson I., Booch G., and Rumbaugh J. 1999. "Unified Software Development Process", Addison-Wesley. 512 p.

Jones. L.G and Lattanze.A.J. 2001. "Using the architecture tradeoff analysis method to evaluate a wargame simulation system: a case study". CMU SEI Technical Report CMU/SEI-2001-TN-022, Software Engineering Institute, Pittsburgh, PA (Dec. 2001). 33 p.

Kazman R, Klein M, Clements P. 2000. "ATAM: method for architecture evaluation". Technical Report CMU/SEI-2000-TR-004, Software Engineering Institution, Carnegie Mellon University. 83 p.

Kerkow D., Kohler K., Dorr J. 2003. "Usability and other quality aspects derived from Use cases». Performance by Design. Proceedings of forUSE 2003, Second International Conference on Usage-Centered, p. 135-154.

Kitchenham B., S. Linkman, A. Paquini, V. Nanni. 1997. "The SQUID approach to defining a quality model", Software Quality Journal, Vol.6, n°3, p.211-233.

Kotonya, G. and I. Sommerville 1996. "Requirements Engineering with Viewpoints". *BCS/IEE Software Engineering Journal,* Vol.11, n° 1, p.5–18.

Lauesen S. 2001. "Software requirements Styles and Techniques". Addison Wesley. 608 p.

Lee.J, Choo Y. 2001. "Quality requirements elicitation for the architecture evaluation of process computer systems". In Proceedings of APSEC. (4-7 December, Macau, China), p. 335-340.

McCall J. A. 1977. "Factors in Software Quality - General Electric," n77C1502.

Maguire M.C. 1998. "User-Centred Requirements Handbook", HUSAT Research institute, WP5, deliverable D5.3 from Telematics Applications Project TE 2010. 202 p.

Mead, N. R. 2004. "Requirements Elicitation and Analysis Processes for Safety & Security Requirements." Proceedings of the Third International Workshop on Requirements for High Assurance Systems (RHAS 2004). Kyoto, Japan, Sept. 6, 2004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/community/rhas-workshop/rhas04-proceedings.pdf. P 35-40.

Mead, N. R. 2005. “Security Quality Requirements Engineering (SQUARE) Methodology.” Technical report CMU/SEI-2005-TR-009. ESC-TR-2005-009. 81 p.

Moreira A., J. Araujo, and I. Brito. 2002. "Crosscutting quality attributes for requirements engineering," in Proceedings of the 14th international conference on Software engineering and knowledge engineering. (Ischia, Italy: ACM Press), p. 167-174.

Mylopoulos J., L. Chung, B.Nixon. 1992. "Representing and Using Non-Functional Requirements: A Process Oriented Approach", ACM Transaction on Software Engineering, vol.18, n°6, pp.483- 497.

Mylopoulos. 1998. “Why Goal-Oriented Requirements Engineering? Proceedings of the 4th International Workshop on Requirements Engineering: Foundations of Software Quality (8-9 June 1998, Pisa, Italy), p. 15-22. E. Dubois, A.L. Opdahl, K. Pohl, eds. Presses Universitaires de Namur, 1998.

NIST (National Institute of Standards and Technology), U.S Department of Commerce Technology Administration. 2002. “The Economic Impacts of Inadequate Infrastructure for Software Testing.” Gaithersburg, Maryland. RTI Project Number 7007.011. http://www.nist.gov/director/planning/upload/report02-3.pdf. Retrieved in 2012.

Ozkaya Ipek, Len Bass, and Robert L. Nord, Raghvinder S. Sangwan, “Making Practical Use of Quality Attribute”. I E E E S o f t wa r e. Vol. 25. n°2 (March-April 2008), p.25-33.

Paech B., A. von Knethen, J. Doerr, J. Bayer, D. Kerkow, A. Trendowicz. T. Punter. 2003. "An Experience-Based Approach for Integrating Architecture and Requirements Engineering". 2nd International Workshop on SofTware Requirements to Architectures (STRAW '03), (Portland, Oregon, May 9), p. 142-149.

Paech B., A. Dutoit, D. Kerkow, A. von Knethen. 2002. "Functional requirements, non-functional requirements and architecture specification cannot be separated – A position paper”. Proceedings of the International Workshop on Requirements Engineering: Foundations for Software Quality (REFSQ). (Essen, Germany), p.102-107.

Pfleeger S.L. 2001. "Software Engineering Theory and Practice". In Prentice Hall. Second edition. 659 p.

Poort E. R. and P. H. N. de With 2004. "Resolving requirement conflicts through non-functional decomposition," presented at Software Architecture, 2004. WICSA Proceedings, Fourth Working IEEE/IFIP Conference. (12-15 June Oslo, Norway), p.145 - 154.

Punter T., R. V. Solingen, et al. 1997. "Software Product Evaluation-Current status and future needs for customers and industry", In the Proceedings of the 4th IT Evaluation (EVIT-97). (Netherlands, Delft), p. 1-11.

Punter T., A. Trendowicz, P. Kaiser. 2002. "Evaluating Evolutionary Software Systems", PROFES 2002 (product focused software process improvement): international conference on product focused software process improvement N°4, (Rovaniemi, 9-11 December Finlande) , vol. 2559, p. 258-272.

A. Rashid, P. Sawyer, A. Moreira, and J. Araujo. 2002. "Early Aspects: a model for aspect-oriented requirements engineering," presented at Requirements Engineering Proceedings, IEEE Joint International Conference on, 2002. (9-13 September Essen, Germany), p. 199 - 202.

Sangwan Raghvinder, Colin Neill, Matthew Bass, Zakaria El Houda. 2008. «Integrating a software architecture-centric method into object-oriented analysis and design" in Journal of Systems and Software. Vol. 81, n° 5, p. 727–746.

Sawyer.P., Sommerville.I. and Viller.S. 1996. "PREview: Tackling the Real Concerns of Requirements Engineering". Cooperative Systems Engineering Group, Technical Reports.

Space-Ufo Consortium. 1998. "The Space Ufo Methodology-User Guide", Esprit project P22290.

Solingen Van, R., R.J. Kusters, J.J.M. 1999b. "Strategies for the identification and specification of embedded software quality". Proceedings: Software Technology and Engineering Practice, STEP'99. (Pittsburgh, PA, USA August 30-September 2), p 33-39.

Sommerville I. and P. Sawyer. 1997. "Requirements Engineering, A good practice guide", John Wiley and Sons. 404 p.

Sousa G., Vastro j. 2004. "Improving the separation of non-functional concerns in requirements artifacts"Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04).

Suryn W., A. Abran. 2003. "ISO/IEC SQuaRE. The 2nd generation of standard for quality of software product". Proceedings of 7th IASTED International Conference on Software Engineering and Applications, SEA 2003, November 3-5, 2003, Marina del Rey, CA, USA. p 807-814.

Suryn W. 2003. "Thoughts on Teaching Software Quality Engineering". Proceedings of 8th Annual INSPIRE Conference (Springer), April 23-25. Galsgow, Scotland, UK.

Suryn W., Abran A., Laporte C. 2004a. “An integrated life cycle quality model for general public market software products”. Proceedings of 12th International Software Quality Management and INSPIRE Conference (BSI). (Canterbury, Kent, UK 5-7 April), p. 97-110.

Suryn W., Hailey V. A., Coster A. 2004b. “Huge potential user base for ISO/IEC 90003 – the state of the art for improving quality in software engineering”. ISO Management System International No.4, July-August 2004.

Suryn W., Gil B. 2005a. "ISO/IEC9126–3 internal quality measures: are they still useful? » HCTII.

Suryn W., Girard D. 2005b. " Suryn-Abran Consolidated Quality Lifecycle (CQL) Model - the Applicative Evolution". BIS 2005. (Poznan, Poland, 20–22 April), p.126–146.

Suryn W., Kahlaoui A., Georgiadou E. 2005c. "Quality engineering process for the Program Design Phase of a generic software life cycle". Proceedings of 13th International Software Quality Management & INSPIRE Conference (SQM 2005). (Gloucestershire, Cheltenham, UK.21–23 March), p. 253–266.

Suryn W. course 2006a. " Ingénierie de la qualité logicielle", at ETS: École de Technologie Supérieure. 2003-2006.

Suryn W. 2006b. http://profs.logti.etsmtl.ca/wsuryn/research/. Retrieved in 2006.

Suryn W. 2006c. http://profs.logti.etsmtl.ca/wsuryn/iqual/. Retrieved in 2006.

Sustar.J and Goldschmidt.I. 2007. "Saving Energy with a Building Automation System". Document ID: CEMC-EMQ-Q307. http://www.esource.com/resource. Retrieved in 2012.

Sutcliffe A, Minocha S. 1998. "Scenario-based analysis of nonfunctional requirements". In: Dubois E, Opdahl AL, Pohl K (eds) Proceedings of the fourth international workshop on requirements engineering: foundation of software quality—REFSQ 98. (Presses universitaires de Namur, Namur), p 219–234.

TL9000. 2001a. Quality Management System Requirements Handbook. Release 3.0, QuEST Forum 2001.

TL9000. 2001b. Quality Management System Measurements Handbook. Release 3.0, QuEST Forum 2001.

Trendowicz Adam and Teade Punter. 2003. "Quality Modeling for Software Product Lines". Proceedings of 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, QAOOSE, Darmstadt, 2003.

Veenendaal Erik.van and Julie McMullan 1997. "Achieving Software Quality". ESSI SCOPE. 208 p.

Vliet, H. 2002. “Software Engineering, Principles and Practice”, Second Edition. John Wiley & Sons, 705 p.

Westerheim, H., Hanssen, G.K. 2005. “The introduction and use of a tailored unified process - a case study”, in *Proceedings. 31st Euromicro Conference on Software Engineering and Advanced Applications*, p 196-203.

Wiegers Karl E. 1999. "Writing Quality Requirements". Process Impact. Retrieved 05MAY08 from: http://www.processimpact.com/articles/qualreqs.html. Published in *Software Development Magazine*, May 1999.

Yuen Tak Yu, Pak-Lok Poon. 2005. “Designing activities for learning software quality practices”, in *Proceedings Fifth International Conference on Quality Software (QSIC 2005. (Melbourne, Australia 19-20 September)*, p333-338.

Zubrow.D. 2004. “Software Quality Requirements and Evaluation, the ISO 25000 Series”. PSM Technical Working Group. Carnegie Mellon University. Pittsburgh, PA 15213-3890. Retrieved 20 may from http://www.psmsc.com/Downloads/TWGFeb04/04ZubrowISO25000SWQualityMeasurement.pdf. 35 p. Retrieved in 2007.