Software Architects' Experiences of Quality Requirements: What We Know and What We Do Not Know?

Maya Daneva[1], Luigi Buglione[2], and Andrea Herrmann[3]

[1] University of Twente, The Netherlands
[2] Engineering IT SpA, Italy
[3] Hermann & Ehrlich, Germany
m.daneva@utwente.nl, luigi.buglione@eng.it,
herrmann@informatik.uni-heidelberg.de

Abstract. [Context/motivation] Quality requirements (QRs) are a concern of both requirement engineering (RE) specialists and software architects (SAs). However, the majority of empirical studies on QRs take the RE analysts'/clients' perspectives, and only recently very few included the SAs' perspective. As a result, (i) relatively little is known about SAs' involvement in QRs engineering and their coping strategies, and (ii) whatever is known mostly comes from small and midsized projects. **[Question/problem]** The question in this exploratory study is how SAs cope with QRs in the context of large and contract-based software system delivery projects. **[Principal ideas/results]** We executed an exploratory case study with 20 SAs in the context of interest. The key results indicate the role SAs play in QRs engineering, the type of requirements communication processes SAs are involved in, the ways QRs are discovered, documented, quantified, validated and negotiated. Our most important findings are that in contract-based contexts: (1) the QRs are approached with the same due diligence as the functional requirements and the architecture design demand, (2) the SAs act proactively and embrace responsibilities over the QRs, (3) willingness to pay and affordability seem as important QRs prioritization criteria as cost and benefits do, and (4) QRs engineering is perceived as a social activity and not as much as a tool and method centric activity. **[Contribution]** The main contributions of the paper are (i) the explication of the QRs process from SAs' perspective, and (ii) the comparison of our findings with previously published results.

Keywords: Quality requirements, Software architecture design, Exploratory case study, Contract-based software development, Empirical research method.

1 Introduction

In the past 15 years, quality requirements (QRs), also referred to as Non-functional requirements (NFRs), became a key sub-field in the Requirements Engineering (RE) discipline. Many approaches to QRs have been designed and a large number of empirical studies has been carried out to evaluate various aspects of these approaches. However, most of these studies have taken the RE perspective exclusively, be it the

J. Doerr and A.L. Opdahl (Eds.): REFSQ 2013, LNCS 7830, pp. 1–17, 2013.

one of RE researchers, of RE practitioners or of business users. Very few studies have included the perspectives of those stakeholders involved in downstream software development activities, e.g. software architects (SAs) and developers. Yet, there is a clear consensus in the RE community that including multiple stakeholders' perspectives is necessary for the field to advance [1]. In particular, the perspective of SAs has been deemed one of the most important given the growing popularity of approaches to joint RE and software architecture [1,2,3,4,5,6,7,8,9]. A 2010 systematic review by Svensson et al [7] identified 229 empirical studies on the various aspects of QRs engineering. However, its authors found only 18 studies being well-documented and stating explicitly the perspectives taken in the empirical research. Moreover, our review of these 18 studies indicated that none of them considered the SAs' perspective. More recently (2010-2012), RE research yielded five publications [6,8,9,10,11] dedicated specifically to the SAs' perceptions on QRs. These studies agree that the perspectives of SAs and RE specialists do differ in both interpreting the role of QRs in downstream software development activities and in reasoning about how QRs problems and solutions are framed, analysed and resolved. While these empirical analyses represent an important progress to narrow the gap in our understanding of how SAs cope with QRs, the experiences these studies report come mostly from small and midsized projects.

In the present paper, we complement these published results with findings from an exploratory study with 20 SAs working in large or very large contract-based projects. The key results of the paper indicate the roles SAs play in QRs engineering, the type of requirements processes SAs are involved in, the way QRs are discovered, documented, quantified, validated and negotiated. The results are compared with findings from prior studies and some implications are drawn for research and practice. In what follows, Sect. 2 is on background and related work and Sect. 3 – on the case study research design. Sect. 4 and Sect. 5 present the results and our discussion of them, respectively. Sect. 6 evaluates validity threats and Sect. 7 concludes.

2 The SAs' Perspective on QRs: Background and Related Work

RE research has generated an overwhelming amount of publications explaining why and how SAs' perspective on QRs is key to project success, e.g. [2,3,4,5,6,7,8]. QRs are deemed key to defining architecture design and many large organizations now organize their work processes on the premise that SAs need to understand the clients' QRs because only through this they can get fully aware of the consequences of their design decisions from clients' perspective (e.g. impacts on total cost of ownership [12]). A 2010 systematic review [13] on the topic of architectural descriptions found a range of QRs driving the final architecture descriptions of those systems reported in published empirical studies. A 2011 SAs' survey [9] stated that SAs are "the population to deal with NFR". These authors went on making the case that shared knowledge and good practices on how SAs address QRs can potentially benefit IT organizations in two tangible ways, (1) by giving them a fact-based ground for improving their SA practices and (2) by helping them devise approaches that make the chances of success more predictable.

As the Introduction mentioned, the SAs' perspective on QRs was the explicit focus of five studies [6,8,9,10,11]. In [9], a survey with one company's SAs uncovered the importance of QRs as perceived by SAs, the ways in which they dealt with QRs, and the impact this had on IT project success. This survey found that projects where modifiability is perceived to be of low business criticality "lead to consistently high customer satisfaction" [9]. Also, projects that used QRs verification techniques were more successful than those that did not. In [11], the authors explain the effect of contractual client-vendor relationship on the interaction of software architecture and quantification of QRs. The authors put forward that if information-sharing between parties is limited, this would significantly impede the quantification of QRs. In [6], QRs were deemed instrumental for the SA´s reasoning process while 'architecting'. Next, a 2010 survey [10] and a follow-up 2012 exploratory interview-based study [8] with SAs in small and medium size projects in Spain, investigated how SAs dealt with QRs. In these project contexts, the findings revealed the state of the art practices regarding QRs elicitation, modelling, and tool support. The study indicated important gaps between the QRs themes currently researched in the RE community and the state-of-the art industry needs, which motivated the authors' call for more empirical research on QRs from SAs' perspective.

3 Goal, Research Questions and Empirical Research Design

The goal of this study is to understand how SAs cope with QRs in large and contract-based software system development projects. Specifically, we wanted to gain insights and answer the following research questions: (RQ1) How do the SAs understand their role? (RQ2) Do SAs and RE staff use different terminology for QRs? (RQ3) How do QRs get elicited? (RQ4) How do QRs get documented? (RQ5) How do QRs get prioritized? (RQ6) How do QRs get quantified, if at all? (RQ7) How do QRs get validated? (RQ8) How do QRs get negotiated? (RQ9) What role does the contract play in the way SAs cope with QRs?

We conducted an exploratory multiple-case study, applying Yin's guidelines [14] and using structured open-end in-depth interviews with practitioners from 20 software project organizations (Table 1). The application domains where the SAs developed software solutions represent a rich mix of fields, incl. telecom, real estate management, air transportation, entertainment (online gaming/video streaming) educational services, hospitality services, and ERP. Our study was performed in four steps: (1) Compose an interview guide following the guidelines in [15]; (2) Do a pilot interview to check the applicability of the guide to real-life context; (3) Carry out interviews with practitioners according to the finalized questionnaire; (4) Sample and follow-up with those participants that possess deeper knowledge or a specific perspective. The interview guide is receivable from the authors. Each interview lasted 35 to 45 minutes. Eight interviews took place face-to-face, and 12 on the phone. Each interviewee was informed on the research purpose and the research process by the first author, and provided with the list of interview questions at least a week before the interview date. All interviewees came prepared to the interview, which was instrumental to the effective use of the interview time slots.

Choosing the Participants: Our selection criterion for participation in the case study was the participant's exposure to the realm of large and contract-based systems delivery. We included 20 SAs from 14 companies in the Netherlands, Belgium, Finland, and Germany. At the time of the interviews, the companies were engaged in large contract-based [1] projects. More in detail, the SAs were selected because they (i) had professional backgrounds pertaining to our research questions and its context (i.e. contract-based), and (ii) had the potential to offer information-rich experiences. Also, they demonstrated an interest in exploring similar questions from their companies' perspectives. All 20 SAs had the following backgrounds:

(1) They all worked in large projects that were running in at least three different development locations in one country, and had clients in more than two countries.

(2) All SAs had at least 10 years of experience in large systems and were familiar with the interactions that happen between SAs and RE staff.

(3) All SAs worked in contract-based projects where contracts between parties were established in two steps [1]: first, a contract was agreed upon for the purpose to get the requirements documented in sufficient detail, so that a SA can use them to work on architecture design. Then, a second contract dealt with the system delivery itself. The pricing agreements varied across the participating companies. Some were fixed-price, others variable, and a third group included a combination between fixed-price and variable. Four SAs worked in outsourcing contracts, and 16 were employed on projects where software development sub-contractors were participating. All SAs deemed their contracts comprehensive and aligned with the spirit of their projects. (In other words, none suggested their organization had any issue with the contract).They also said that their projects were considered successful, by both parties in the contract. The SAs got to know the first author during various business and research conferences in the period of 2001-2012. Using purposive sampling [15], she chose the interviewees, based on her knowledge about their typicality. The number of participants was large enough to provide a broad variety of viewpoints. We planned the interviews to be 'structured' [15] with regard to the questions being asked during the session. This means, the interviewer was the one to control what topics would be discussed and in which order.

We note that interview-based exploratory case studies usually are intended to promote self-disclosure and that is what we were after in this work. We collected data via one-on-one interactions of a researcher with each of the interviewees that have various backgrounds but also common professional values and common roles in which they execute their professional duties. As in [15], interview studies are not used to provide statistically generalizable results applicable to all people similar to the practitioners in a specific study. The intention of the exploratory case study is not to infer, but to understand and not to generalize, but to determine a possible range of views. Therefore, in this study we will adopt, based on the recommendations in [15], the criterion of transferability as a useful measure of validity. Transferability asks for whether the results are presented in a way that allows other researchers and practitioners to evaluate if the findings apply to their contexts.

Table 1. Case study participants and organizations

ID	Business	System description	Team size (# of people)	Project duration (months)
P1.	Large IT Vendor	ERP package implementation (Oracle)	35	18
P2.	Large IT Vendor	ERP package implementation (SAP)	60	15
P3.	Large IT Vendor	ERP package implementation (SAP)	75	18
P4.	Large IT Vendor	ERP package implementation (SAP)	41	12
P5.	Large IT Vendor	ERP package implementation (SAP)	51	12
P6.	Large IT Vendor	ERP package implementation (Oracle)	45	12
P7.	IT Vendor	ERP package implementation (SAP)	40	18
P8.	Software Producer	Online learning environment	22	12
P9.	Software Producer	Sensor system for in-building navigation	35	12
P10.	Software Producer	Online ticket booking application	15	12
P11.	Oil & Gas	Logistics planning application	21	12
P12.	Insurance	Web application for client self-service	61	24
P13.	Insurance	Client claim management and reimbursement app	53	16
P14.	Real Estate	Web application for rental contract handling	42	18
P15.	Air Carrier	Web app for passengers' feedback processing	11	14
P16.	Video Streaming	Viewer recommendation management system	18	18
P17.	Video Streaming	Viewer complaint management system	45	9
P18.	Online bookstore	Order processing system	15	10
P19.	Online game producer	Gaming system	81	21
P20.	Online travel agency	Room deal identification system	45	12

Data Analysis Strategy: We were guided by the Grounded Theory (GT) method of Charmaz [16], which is a qualitative approach applied broadly in social sciences to construct general propositions (called a "theory" in this approach) from verbal data. GT is exploratory and well-fitting situations where the researcher does not want to use pre-conceived ideas, and instead is driven by the desire to capture all facets of the collected data and to allow the propositions to emerge from the data. In essence, this was a process of making analytic sense of the interview data by means of coding and constant comparison of pieces of data that were collected in the case study. Constant comparison means that the data from an interview is constantly compared to the data already collected from previously held interviews. We first read the interview transcripts and attached a coding word to a portion of the text – a phrase or a paragraph. The 'codes' were selected to reflect the meaning of the respective portion of the interview text to a specific research question. This could be a concept (e.g. 'willingness to pay'), or an activity (e.g. 'operationalization', 'quantification'). We clustered all pieces of text that relate to the same code in order to analyze it in a consistent and systematic way. The results of the data analysis are in Sect. 4 and the discussion on them – in Sect. 5.

4 Results

Our findings are presented as related to each research question. As it is usual in qualitative studies, we supplement the observations with interviewees' quotations.

4.1 RQ1: How Do the Software Architects Understand Their Role?

All interviewees indicated their companies had SAs' job descriptions that list their most important duties, skills competence levels, and job salary scales. As all the organizations were mature in terms of project management processes and process-oriented thinking, the roles of the SAs were established and they were clearly recognizable by their fellow team members. 13 out of the 20 SAs thought of their role as *'a bridge'* between QRs and the underlying technology that they were dealing with.

"You've got to translate what their clients want in terms of working solutions and technology choices" (participant P1), "*You learn pretty quickly to stretch the technology so that it works for those who you call 'requirements engineers', I mean here our Business Analysts and their "patrons", the business unit directors" (participant P2),*

"I have days when I do nothing but 'translating' the language of our millions of players into the features that are technologically implementable in the next release. All I do is walking through a list of "user experiences" that our requirements folk want to launch in the market, and I make sure that it all works well at implementation level, that features build upon each other, so that the overall gaming experience and enjoyability are never compromised. That's why I'm coming to work every day" (participant P19).

Other seven SAs thought of their roles as *'review gate keepers'* in light of the active roles in QRs reviews, contract compliance evaluation and sign off:

"If you are sick, on vacation, or on a conference, there is no way this can go unnoticed; they will need you and if there is no one to tell them that what they are doing is right [according to contract and project goals], I mean it pure architecturally; they are going to wait for me so that I give the architecture approval. Otherwise, we run a risk of screwing up things miserably" (participant P15).

How many SAs are enough for engineering QRs? All SAs experienced one or two SAs have been involved in their projects. In case of two SAs to be involved, our interviewees meant one SA from the vendor and the client side, each.

"You may work with many RE people for the different subject areas, but one architect must be there to aggregate their input and evaluate what it means for the underlying architecture" (participant P6).

4.2 RQ2: Do SAs and RE Staff Use Different Terminology for QRs?

All SAs considered the process of gaining communication clarity with RE staff a non-issue. They thought this was due to their long years' experience in their respective business sectors (all interviewees had 10+ years of experience as indicated in Sect 3). Even when working with less experienced RE specialists, they thought that their domain knowledge has been instrumental to spot missing or incomplete requirements. For example, a SA who worked on the development of an online system for processing an air carrier's clients' feedback said that in this application domain scalability is usually regarded as the QR of the highest priority. If a specification says nothing about it, he considers this a *'red flag'* and acts accordingly:

"If your RE person says nothing about it, you are absolutely sure there is something going wrong here. And you better pick up the phone and call them ASAP because they may be inexperienced and who knows what else could go wrong beyond this point. Not doing this is just too much risk to bear" (participant P15).

Those SAs working in ISO-certified organizations suggested that the knowledge of the ISO standards adopted in their company, the mandatory ISO-training that everyone in the IT department should go through, *'the habit of looking back to what the ISO-compliant Quality Manual says'* help tremendously both the SAs and RE-staff understand each other: *"We don't use the term QRs, not even non-functional requirements, we call them ISO aspects, because we are ISO-certified and our system must show compliance to the ISO standards. Also our requirements specialists have in their template a special section that contains these aspects. We enumerate them one after another. All relevant aspects must be there, security, maintainability, performance, you name it. It's more than 40 aspects [in the company's Quality Manual that is consulted on an on-going basis]. They [the RE staff] know them and we know them [because we went to the same training], so we have a common ground" (participant P11).*

An interesting observation was shared by those SAs delivering large SAP systems. They indicated, the SAP vendor's Product Quality Handbook included around 400 QRs which are implemented in the standard software package and which everyone on the development team is aware of (as these team member learnt about the Product Quality Handbook in their professional training). If there were QRs specific to the client and unaddressed in the Handbook, then those should be specified on top of the 400 that come in the SAP's package. (SAP is ISO 9001-certified).

4.3 RQ3: How Do QRs Get Elicited?

14 SAs used checklists. These were based on a variety of sources: (i) ISO standards (e.g. 25045-2010, 25010-2011, 25041-2012, 25060-2010), (ii) architecture frameworks, be they company-specific or sector-specific, (iii) internal standards (e.g. vendor/client-organization-specific), and (iv) stakeholder engagement standards, e.g. AA1000SES [17]. Regardless the source, the interviewees agreed that the checklist-supported process is always iterative, because not all QRs could get 100% clear at the same time. *"You've got to go at least 4-5 times until you get a spec that makes sense to me and my fellows" (participant P4).*

In their views, the fact that most QRs are not global to the system but act on specific pieces of functionalities, imposes an order in the elicitation activities: SAs do expect first the functional requirements to be elicited and then to use the resulting functional requirements specification as the ground for eliciting QRs.

"*How otherwise will you know which attribute matters to what piece of functionality?" (participant P9).*

Four SAs argued QRs are never elicited but detected, e.g. one SA was involved in experimental serious-game-based process specifically designed to "detect" QRs. Two others shared they had *"a bunch of pilot users who volunteer to play the role of guinea pigs; they are passionate about the system and would love to tell you where it*

fails to deliver up to their expectations in terms of performance, availability, and user experience" (participant P10). Another SA had been using storytelling techniques to uncover QRs, together with his RE counterpart and his clients.

4.4 RQ4: How Do QRs Get Documented?

15 out of the 20 SAs specified QRs by using predefined templates. Some of them were vendor-specific, e.g. in SAP projects, SAs used SAP's standard diagram notation called Technical Architecture Modelling [18], as it has been part of the SAP Architecture Curriculum [12]. Others were derived based on (i) the ISO standard, (ii) the House of Quality (HoQ) of the Quality Function Deployment (QFD) methodology [19], (iii) the Planguage approach [20], and (iv) the INVEST grid approach [21].

The other 5 SAs were using plain natural language text that provides at least the definition of each QR, plus information on the end user to do the acceptance test on it and the ways to demonstrate that the system meets it. The amount of detail in these specifications varied based on what the SAs deemed important to be provided to them by the RE staff or the users. For example, one SA wanted to hear a story from a user on *"how the user will know the system is slow? A user can tell you "If I manage to get one of my phone calls done while waiting, this means to me it's very slow." (participant P12).* Other SAs said they write their QRs definitions next to the functional requirements, if these are specified in a process model or a data model. *"This way you will know which smallest pieces of functionality and information entities are affected by which QR" (participant P1).*

4.5 RQ5: How Do QRs Get Prioritized?

All SAs agreed that (i) they make QRs trade-offs as part of their daily job on projects, and (ii) the project's business case was the driver behind their trade-off decision making. The key prioritization criteria for making the QRs trade-offs were cost and benefits, evaluated mostly in qualitative terms but whenever possible also quantitatively, e.g. person-months spent to implement specific QRs. Perceived risk was identified as subsumed in the cost category, as SAs deemed a common practice the tendency to translate any risks to QRs, into costs to bear in a contractual agreement. However, next to perceived cost and benefits, 12 SAs put forward two other QRs prioritization criteria: *client's willingness to pay* and *affordability*. The first is about the flexibility of the value of a QR to the client. It is expressed as the level of readiness of client organizations to pay extra charges for some perceived benefit that could be brought by implementing a specific QR. Six SAs elaborated that this criterion alone is instrumental to split up the QRs in three groups: (i) essential, which includes those QRs that directly respond to the reason of why the client commissioned the system in the first place, e.g. in a hotel reservation system, it's absolutely essential that a secure payment processing method is provided; (ii) marginal, which includes those QRs that are needed yet clients are willing to spend little on them, e.g. a user interface feature that might be appreciated and would be perceived as a value to pay a few hundred euro, but not thousands of euro; and (iii) optional, which includes QRs

that clients will find enjoyable to have but would not be willing to pay for, e.g. flashy animation effects in a game system that are fun, yet not truly a *'money-maker'*.

Next, affordability is about whether or not the cost estimation of a QR is in accord with (1) the resources specified in the contract and (2) with the long term contract spendings of the client organization. This criterion determines whether or not a QR is aligned with (short term) project goals and/or (long-term) organizational goals.

Furthermore, SAs' experiences differed regarding who ultimately decides on the QRs priorities. 13 SAs suggested that in their projects the prioritization has been linked to a business driver or a KPI in the project and it has been the project's steering committee to decide on the priorities. They deemed the prioritization process *'iterative'* and a *'learning experience'* in which the SAs learn about what QRs the drivers *'dictate and how desperately the company needs those QRs'* and the RE specialists and the business owners learn about the technical and cost limitations.

"It's through this mutually educational process that you arrive at the priorities. This takes a few iterations, starting always from the Steering Committee that tells us what's most important to them. We tell them what we can do and check against budget, timelines and people resources. If we cannot do it within our limits, then they must decide either to pour more money into the project, or to rethink what they could live without, so that we have a good enough solution for our circumstances". (participant P11).

In contrast to these 13 SAs, the other 7 considered themselves as the key decision-makers in setting the priorities:

"You can stretch a technology to a certain point. No matter what your client wants, once in a while you've got to say 'no' and push back decisively" (participant P11).

Concerning the use of requirements prioritization methods, 19 SAs suggested no explicit use of any specific method other than splitting up QRs in categories of importance, namely 'essential', 'marginal', and 'optional'. SAs named these categories differently, yet they meant the same three. One SA named EasyWinWin [23] as the group support tool aiding the requirements prioritization and negotiation activities in a project. Three out of the 19 shared that nailing down the top 2-3 most important QR is a non-issue, because of the obviousness of these requirements to the client, e.g: *"If you develop a game, it's all about scalability and user experience. You have no way to make your fancy animation work if you cannot get scalable and if you jeopardize user experience" (participant P19).*

What is an issue is the prioritization of those QR that take less prominent place from user's perspective, e.g. maintainability, evolvability, e.g:

"These requirements matter to you, not to the client. Even you do a brilliant job on maintainability, nobody will pat you on the back and say thank-you for this. It's very difficult, I'd say, almost political, to prioritize this kind of QRs" (participant P13).

4.6 RQ6: How Do QRs Get Quantified, If at All?

All SAs agreed that expressing QRs quantitatively should not happen very early in a contract-based project. This was important in order to prevent early and not-well-thought-out commitments. Three SAs said they rarely use quantitative definitions of

QRs. Instead, they get on board an expert specialised in a specific QR (e.g. usability, or scalability) and let him/her *'do the quantification job'* for them. 10 other SAs used as a starting point the quantitative definitions that were pre-specified in the contract (e.g. a contract may state explicitly that the system should scale up to serve hundreds of thousands of subscribers). However, 8 of the 10 warned that more often than not contracts address design-level requirements [1], including (i) detailed feature specifications of how QRs are to be achieved (e.g. specifying a proprietary influence metric to be included in the ranking algorithms of the recommender system that is part of a larger online video streaming system), or (ii) a particular algorithm rather than required quality attribute value and criteria for verifying compliance (e.g coding a very specific search algorithm for finding hotel deals). Confusing QRs with design-level requirements was deemed a critical issue in industry; it points out to a mismatch in understanding what is really quantified and by using what kind of measures: design-level requirements are quantified by using product measures and not project measures which are those important for contract monitoring purposes. However, more often than not contracts use product and project measures incorrectly, the final effect being that a number of project tasks related to implementing QRs don't get 'visible' but 'implicit', and therefore no budget is previewed for them and, in turn, the client would not commit to pay.

Next, 7 SAs worked with a team of systems analysts on operationalizing QRs. In essence, it meant decomposing them until reaching the level of architecture design choices or of the smallest pieces of functional requirements. Once at this level, the practitioners felt comfortable starting quantifying the QRs, e.g. they used the operationalization specifications as input to a Function Points counting process, in order to 'size the QRs'. However, no common quantification method was observed to be used. Instead, the SA suggested the choice of a method should match the project goal (i.e. towards what end quantification was needed in the first place). E.g., if quantification of QRs is to serve project management and contract monitoring purposes, then it might be well possible that in the future those organizations experienced in Function-Points-based project estimation, would use the newly released IFPUG NFR Assessment standard [22], called SNAP (Software Non-functional Assessment Process).

4.7 RQ7: How Do QRs Get Validated?

All SAs were actively involved in QRs validation, 16 considered it part of their job, while four said that it's the job of the RE staff to ensure QRs are validated. These four SAs used the RE specialists as contact points on clarifying requirements. We make the note that for the purpose of this research, we call 'validation' the process that (a) ensures that QRs clearly describe the target solution, (b) confirms these QRs are technically implementable and the resulting architecture design satisfies the business requirements as per the contractual agreement.

14 SAs participated in requirements walkthroughs with clients led by a RE specialist where clients confirm the functionalities on which the QRs in question were supposed to act. The walkthroughs were deemed part of the client expectation

management process that the project manager established. The SAs considered them as the opportunity to inform the clients about those QRs that could not be implemented in the system or could not be implemented in the way the client originally thought: *"You've got to educate them on what your technology can and cannot do for them and the walkthroughs is how this happens relatively easily."(participant P1).*

Three SAs used the HoQ [19] to demonstrate the strength of the relationship between a QR-statement and its operationalization in terms of either functional requirements or architecture design choices. Two SAs validated QRs against internal architecture standards. Should they identify deviations from the standards, they escalate this to both managers and RE-staff. In extreme cases, when QRs are grossly misaligned with the architecture standards, this should be brought to the attention of the steering committee, the program director, and the architecture office manager responsible for the project. *"You have to inform them immediately that things have no way to work as planned, so they get back to negotiation and revise the concept" (participant P8).*

RQ8: How Do QRs Get Negotiated?

Ten SAs used their project's business cases as the vehicle to negotiate requirements. They considered this a common practice in enterprise systems projects. *"You need to express yourself in money terms, that they [the clients] can understand very well". (participant P20).*

Three SAs who worked on projects where user experience was the most important QR, said their goal in QRs negotiation is to prevent the most important QR from becoming suboptimal, if other QRs take more resources and attention. These SAs did not use their business cases, but considered effort and budget allocation as important inputs to negotiation meetings.

Five other SAs thought of themselves as *'information providers and mentors'* to the team, but not *'truly negotiators on QRs*' and that it's the project manager's responsibility to lead in the negotiation:

"It's his job to sell it to the other parties. I'm just an internal consultant; what they do with my information is their business" (participant P10).

Other three SAs used the HoQ, EasyWinWin [23], and the Six-Thinking-Hats method [24] to reason about QRs in negotiation meeting, respectively. We note that the Six-Thinking-Hats is a general approach to resolving complex issues and companies use it for any negotiation situation, be it QRs related or not. The approach was well-received and internalized in the company and people *'had fun using it as it takes pressure off them in this kind of difficult conversations' (participant P16).*

RQ9: What Role Does the Contract Play in the Way SAs Cope with QRs?

Did SAs have to refer to the contract, enforce it, or use it in any way in their projects so far? In the SAs' experiences, there were three ways in which the contract influenced how they coped with QRs: (1) the contract enforced the cost-consciousness

of the SAs and was used to evaluate the cost associated with achieving the various QRs; (2) the contract stipulated QRs levels, e.g. in the Service Level Agreement (SLA) part, that were targeted and subjected to discussions with the stakeholders; and (3) the contract in fact pre-defined the priorities for some small but very important set of QRs. 17 SAs indicated that the contract was used on an on-going basis to stay focused on what counts most in the project. To these SAs, the contact was the vehicle to ensure the system indeed includes *'the right things', 'those that they needed in the first place, and that we are billing them for'*.

12 SAs shared that a contract-based context is conductive to understanding QRs as a way to maintain control. In their views, every comprehensive contract usually comes with SLA specifications, key performance indicators (KPI) and measurement plans that address multiple perspectives (clients/vendors), e.g., the Balancing Multiple Perspectives technique [25] is a way to help all involved parties in understanding and validating the right amount of things to do in a contract-based project.

In contrast to this, three SAs thought the contract was not that important. They said, it was just *'the beginning of their conversation'* on QRs and not as the reference guide to consult on an on-going basis. They thought it's the people who make the contract work. In their view, it has always been the RE-staff and project managers who work with the contract and who usually communicate to everyone else if the project efforts get misaligned with the clauses of the contract.

5 Discussion

This section compares and contrasts our finding to those in previously published studies on QRs from SAs' perspective. Our discussion is organized according to our research questions.

RQ1: How do the SAs understand their role? Our results suggest that in contract-based and large projects, the SAs define their role as "a bridge" that connects clients QRs to the architecture design. This contrasts to the results in [8] where the SAs had indicated a broad diversity of roles and tasks they took on (e.g. coding). We think the contrast is because our SAs came from regulated environments where terminology, roles and processes are determined, well communicated, and lived up to. Our findings agree with [11] on the importance that SAs place on gaining as deep as possible understanding of the QRs and using it to deliver a good quality architecture design. Regarding how many SAs are enough for engineering QRs, we note that this question has not been yet researched in RE studies. In our interviewees' experiences, it's usually one or two SAs that operate together in a large contract-based project. This is in line with Brooks' most recent reasoning on "the design of design" of large systems [26] where he explains that a 2-person team can be particularly effective where larger teams are inefficient (other than for design reviews where the participation of a large number of reviewers is essential).

RQ2: Do SAs and RE staff use different terminology for QRs? Our results did not indicate this as the issue that preoccupied the SAs. This contrasts [8] where Spanish SAs collectively indicated a broad terminological gap between SAs and RE staff. The

authors of [8] made observations that the absence of shared glossary of QRs types was part of the problem. We think the difference between our findings and those in [8] may well be due to the fact that our case study projects were happening in regulated organizations where standards defined terminologies that all project team members adopted (including SAs) and assumed ownership over their use. We found that SAs implicitly referred to at least two main streams of standards: (i) management systems as e.g. 9001-27001-20000, and (ii) 'technical standards' (as 14143-x, 9126-x, etc.), where (i) are about 'requirements' to be accomplished and (ii) are about processes and/or solutions about 'how to' do things. However, terms from both streams were used interchangeably and it was not clear which QR followed the terminology of which stream. This made us think that contract-based development would greatly benefit if a common and shared glossary of ISO terms existed (and help make explicit the difference between the two steams of ISO standards).

RQ3: How do QRs get elicited? Our study revealed all SAs were actively involved in elicitation. This agrees with [9] and is in contrast to [8]. Our assumption is that the sense of ownership over the QRs that the SAs shared could be the possible reason for their proactiveness and involvement. Also, we found checklist-based techniques as the predominant approach in contract-based project context.

RQ4: How do QRs get documented? We found that standardized forms/templates plus natural language were used most. This is in contrast with [8] where the SAs could not agree on one specific systematic way to document QRs and natural language was the only common practice being used. Why this difference occurs? We assume that it's because of the regulated nature of the contract-based environments and of the use of standards. We think it's realistic to expect that in such contexts, a contract is monitored on an on-going basis (e.g. all relevant SLAs are well specified and how they would be measured is explicitly defined) and its use forces IT professionals to adopt a sound template-based documentation flow throughout the project [27].

RQ5: How do QRs get prioritized? Two 'new' prioritization criteria crystalized in this study: client's willingness to pay and affordability. These complement the well-known criteria of cost, benefits and risk (as stated e.g. in [28]). We assume that the choice of these criteria could be traced back to the contract-based nature of the projects in our study where both vendors and clients had to get clear as early as possible on, scope, project duration and the way they organize their work processes and its impact on each party.

RQ6: How do QRs get quantified, if at all? Our results agree with [11] on the importance of QRs quantification in practice. However, unlike [4], our SAs did not indicate that searching for new or better quantification techniques was their prime concern. Similarly to [11], they warned about the pitfalls of premature quantification, meaning that early QRs quantification may be based on too many assumptions about the solution. As in [11], our SAs thought that if those assumptions turn out unrealistic, a vendor may find itself in the precarious situation of having committed resources to unachievable QRs. Regarding how quantification happens, the SAs suggest that either using a standard (e.g. 22]) or engaging an expert in specific type of QRs (e.g. security, scalability). While the first ensures that all tasks of implementing QRs are explicitly

accounted for in a project, the second allows for deeper analysis on a single quality attribute and its interplay with others.

RQ7: How do QRs get validated? No SA witnessed a contract that stated an automated (model-checking) tool be used for validating QRs. Instead, our results suggest that common sense practices dominate the state-of-the-art:e.g. using requirements walkthroughs, documentation reviews, building up communication processes around the artefact-development activity (e.g. escalation if a QR is not timely clarified) are simple, yet powerful ways to ensure QRs are aligned with client's expectations and SLAs. This contrasts the QRs literature [4,7] where much accent is placed on tools and methods. One could assume that the active and persuasive behaviour of the SAs regarding QRs validation could be due to the explicit contractual agreements (e.g. SLA), controls and project monitoring procedures (e.g. KPI).

RQ8: How do QRs get negotiated? The business case turned out to be the most important vehicle for SAs to support their negotiation positions in meetings with other stakeholders. In contrast to RE literature (e.g. [1]) that offers an abundance of negotiation methods, we found that only one (EasyWinWin) was mentioned (and it's by only one SA). SAs hinted to general purpose negotiation techniques, e.g. the Six-Thinking-Hats method as being sufficient to help with QRs negotiation. This suggests that it might be worthwhile exploring the kind of support that negotiation methods from other fields (management science, psychology) can offer to QRs negotiation.

RQ9: What role does the contract play in the way SAs cope with QRs? We found that the contract reinforced the SA's role in his/her project organization and redefined the inclusion of this role in QRs engineering. We observed, the SAs were well aware of the possible impacts of a contract on their professional behaviour, e.g. SAs assumed responsibility to clarify QRs priorities and escalated to project managers and/or RE staff if they suspected project goals threatened or contact clauses being violated. Because we could find no prior study that looked into how contracts shape the professional behaviour of RE professionals or of SAs with respect to how QRs are dealt with, we consider it an interesting line of future research.

6 Validity Threats

Our evaluation of the possible threats to validity of the observations and conclusions in this research, followed the checklist in [29]. As our research is exploratory, the key question to address when evaluating the validity of its results, is [15]: to what extent can the practitioners' experiences in coping with QRs could be considered representative for a broader range of projects, companies, application domains? Our case study projects are not representative for all the possible ways in which engineering of QRs is performed in large contract-based settings. Following [30], we think that it could be possible to observe similar experiences in projects and companies which have contexts similar to those in our study, e.g. where large and contract-based projects hire experienced SAs in teams with mature process-oriented thinking, and where standards define the terminology to use for QRs. As the authors of [30] suggest "if the forces within an organization that drove observed behaviour are likely to exist

in other organizations, it is likely that those other organizations, too, will exhibit similar behaviour" (p.12). Moreover, we acknowledge that the application domain may have influenced the ways the SAs coped with QRs. We, therefore, think that more research is needed to understand the relationship between application domains and the way in which the QRs processes happen.

We also acknowledge the inherent weaknesses of interview techniques [15]. A threat is the extent to which the SAs answered our question truthfully. We took two steps to minimize this threat by (i) recruiting volunteers, under the assumption that if a practitioner would not be able to be honest, he/she could decline his/her participation at any stage of the research and (ii) that we ensured no identity-revealing data will be used in the study. Next, it's possible that an interviewee has not understood a question. However, we think that in our study, this threat was reduced, because the interviewer used follow-up questions, and asked about the same topic in a number of different ways. Next, we accounted for the possibility that the researcher might instil his/her bias in the data collection process. We followed Yin's recommendations [6] in this respect, by establishing a chain of findings: (i) we included participants with diverse backgrounds (i.e. industry sector, type of system being delivered), and this allowed the same phenomenon to be evaluated from diverse perspectives (data triangulation [15]); (ii) the interview answers were sent to each SA prior to data analysis to confirm the information he/she provided; and (iii) we had the draft case study report reviewed by the SAs (some of whom provided feedback, which however did not affect our conclusions).

7 Conclusion

The number of studies on the involvement of SAs in QRs engineering is growing and understanding the ways it works would give us a firmer ground for organizing eventually better RE processes that lead to better definitions of QRs, and in turn, architecture designs better aligned with them.

Our study clearly indicates that making SAs an integral part of the QRs processes has been commonplace in the investigated contract-based development contexts. Unlike prior research [8] that revealed important discrepancy between literature and practice as part of small and midsized projects, this study shed light into the perspective of SAs on QRs as treated in large and contract-based projects. Compared to literature, the overriding messages of this paper are that in contract-based projects:

- the QRs are approached with the same due diligence as the functional requirements and the architecture design demand,
- the relationship between RE staff/clients and SAs is actively managed whereby the contract means embracing responsibilities over QRs (and not abdicating thereof),
- client's willingness to pay and affordability seem as important prioritization criteria for QRs as cost and benefits are,
- both the prioritization decisions on QRs and the QRs trade-offs are aligned with the contract (specifically, with the SLA and KPIs),

- checklist-based elicitation is the most common technique for eliciting QRs,
- template-based documentation in natural language is most common approach to documentation,
- quantification of QRs is done with and by experts specialized in a specific QRs sub-field (e.g. performance, scalability)
- QRs validation and negotiation are considered more organizationally and in terms of social interactions with RE staff and clients, than in terms of tool-supported processes.

Our findings have the following implications: To SAs, this study suggests the conversation on QRs starts with the contract, and specifically with the SLA and the business case. To RE tool vendors, it suggests they are better off to think about how RE tools should be better embedded into social processes and broader social interaction context. To RE researchers, our study suggests that instead of solely focusing on QRs methods, tools and techniques, it makes good sense to extend existing research by including analysis of QRs processes as socially constructed ones. How a contract shapes the behaviour of RE staff and SAs is an interesting question demanding future research. One could think of borrowing theories from other disciplines (e.g. behaviour science) to explain why RE staff and SAs engineer QRs the way they do.

In our immediate future, we plan to use these results in follow-up studies, in other countries, specifically in USA, Canada, India, Israel, Brazil and Argentina. Comparing the present results with findings in locations outside Europe would benefit both outsourcing vendors and clients in informing them on the extent to which handling QRs depends on country-specific business culture concerning contractual agreements, industry sector, and levels of maturity.

References

1. Lauesen, S.: Software requirements: Styles and techniques. Addisson-Wesley (2002)
2. Sommerville, I.: Integrated Requirements Engineering. IEEE Software 22(1), 16–23 (2005)
3. Avgeriou, P., Grundy, J., Hall, J.G., Lago, P., Mistrík, I. (eds.): Relating Software Requirements and Architectures. Springer (2011)
4. Capilla, R., Babar, M.A., Pastor, O.: Quality requirements engineering for systems and software architecting: methods, approaches, and tools. Requir. Eng. 17(4), 255–258 (2012)
5. Bass, L., et al.: Software Architecture in Practice, 2nd edn. Addison-Wesley (2003)
6. van Heesch, U., Avgeriou, P.: Mature Architecting - A Survey about the Reasoning Process of Professional Architects. In: 9th WICSA, pp. 260–269
7. Bentsson-Svensson, R., Höst, M., Regnell, B.: Managing Quality Requirements: A Systematic Review. In: EUROMICRO-SEAA 2010, pp. 261–268 (2010)
8. Ameller, D., Ayala, C., Cabot, J., Franch, X.: How do software architects consider non-functional requirements: An exploratory study. In: RE 2012, pp. 41–50 (2012)
9. Poort, E.R., Martens, N., van de Weerd, I., van Vliet, H.: How Architects See Non-Functional Requirements: Beware of Modifiability. In: Regnell, B., Damian, D. (eds.) REFSQ 2011. LNCS, vol. 7195, pp. 37–51. Springer, Heidelberg (2012)

10. Ameller, D., Franch, X.: How Do Software Architects Consider Non-Functional Requirements: A Survey. In: Wieringa, R., Persson, A. (eds.) REFSQ 2010. LNCS, vol. 6182, pp. 276–277. Springer, Heidelberg (2010)
11. Poort, E.R., Key, A., de With, P.H.N., van Vliet, H.: Issues Dealing with Non-Functional Requirements across the Contractual Divide. In: WICSA/ECSA 2012, pp. 315–319 (2012)
12. Groene, B., et al.: Educating Architects in Industry - The SAP Architecture Curriculum. In: 17th IEEE Int. Conf. on Eng. of Computer Based Systems (ECBS), pp. 201–205.
13. Guessi, M., Nakagawa, E.Y., Oquendo, F., Maldonado, J.C.: Architectural description of embedded systems: a systematic review. In: ACM SIGSOFT ISARCS 2012, pp. 31–40 (2012)
14. Yin, R.K.: Case Study Research: Design and Methods. Sage, Thousand Oaks (2008)
15. King, N., Horrock, C.: Interviews in Qualitative Research. Sage, Thousand Oaks (2010)
16. Charmaz, K.: Constructing Grounded Theory. Sage, Thousand Oaks (2007)
17. Stakeholder Engagement Standard (AA1000SES), http://goo.gl/fRopv
18. Groene, B.: TAM – The SAP Way of Combining FCM and UML, http://goo.gl/FW1IA (last viewed on November 8, 2012)
19. Gilb, T.: Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage, Butterworth (2005)
20. Karlsson, J.: Managing software requirements using quality function deployment. SQJ 6(4), 311–326
21. Buglione, L.: Improving estimated by a four pieces puzzle, IFPUG Annual Conference (May 2012), http://goo.gl/bFwRB
22. IFPUG, Software Non-functional Assessment Process (SNAP) – Assessment Practice Manual (APM) Release 2.0 (January 22, 2013)
23. Boehm, B., Grunbacher, P., Briggs, R.O.: EasyWinWin: A Groupware-Supported Methodology for Requirement Negotiation. In: 9th ACM SIGSOFT FSE, pp. 320–321 (2001)
24. de Bono, E.: Six Thinking Hats. Little, Brown, & Co., Toronto (1985)
25. Buglione, L., Abran, A.: Improving Measurement Plans from multiple dimensions: Exercising with Balancing Multiple Dimensions - BMP. In: 1st Workshop on Methods for Learning Metrics, METRICS 2005 (2005)
26. Brooks, F.P.: The Design of Design: Essays from a Computer Scientist. Addison-Wesley (2010)
27. Nicholson, B., Sahay, S.: Embedded Knowledge and Offshore Software Development. Information and Organization 14(4), 329–365 (2004)
28. Herrmann, A., Daneva, M.: Requirements Prioritization Based on Benefit and Cost Prediction: An Agenda for Future Research. In: RE 2008, pp. 125–134 (2008)
29. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14(2), 131–164 (2009)
30. Seddon, P., Scheepers, P.: Towards the improved treatment of generalization of knowledge claims in IS research: drawing general conclusions from samples. EJIS, 1–16 (2011)