Requirements are Slipping Through the Gaps - A Case Study on Causes & Effects of Communication Gaps in Large-Scale Software Development

Bjarnason, Elizabeth; Wnuk, Krzysztof; Regnell, Björn

Published in:
IEEE International Requirements Engineering Conference

2011

Requirements Are Slipping Through the Gaps -

A Case Study on Causes & Effects of Communication Gaps in Large-Scale Software Development

Elizabeth Bjarnason, Krzysztof Wnuk and Björn Regnell
Department of Computer Science, Lund University, Lund, Sweden
E-mail: {Elizabeth.Bjarnason, Krzysztof.Wnuk, Bjorn.Regnell}@cs.lth.se

Abstract Communication is essential for software development as its efficiency throughout the entire project life-cycle is a key factor in developing and releasing successful software products to the market. This paper reports on findings from an explanatory case study aiming at a deeper understanding of the causes and effects of communication gaps in a large-scale industrial set up. Based on an assumption of what causes gaps in communication of requirements and what effects such gaps have, a semi-structured interview study was performed with nine practitioners at a large market-driven software company. We found four main factors that affect the requirements communication, namely scale, temporal aspects, common views and decision structures. The results also show that communication gaps lead to failure to meet the customers' expectations, quality issues and wasted effort. An increased awareness of these factors is a help in identifying what to address to achieve a more efficient requirements management, and ultimately more efficient and successful software development. By closing the communication gaps the requirements may continue all the way through the project life-cycle and be more likely to result in software that meets the customers' expectations.

Keywords:requirements communication; explanatory case study; large-scale requirements engineering; empirical study

I. INTRODUCTION

The requirements communication starts with the customer and continues throughout a development project, involving many different roles. The initially elicited requirements need to be communicated, and changes to those requirements negotiated and communicated between all affected roles, e.g. requirements engineers, developers, and testers. Since change occurs throughout the project, requirements communication must also continue during the entire life cycle [2]. For a software project to be successful, methods and tools must be supplemented with interpersonal communication across functional boundaries, but this needs to be balanced with cost and effectiveness of such communication [12]. Despite this, the bulk of RE processes and research is mainly concerned with requirements in the early project phases, while the ultimate goal of any software project is to efficiently produce successful products; the requirements are just a means to an end. Already in the 1970s, the problem of inefficient and incorrect communication, increased as the requirements ripple through a project involving more people, was reported to lead to overly complex and badly functioning systems [7]. Studies [6], [13], [10], [20] have shown that many of the RE challenges facing large-scale software development are of an organizational and social character, rather than technical, and that projects need to be organized to ensure co-ordination and communication of requirements from marketing to engineering.

Previous studies on communication focus mainly on communication paths [15], [16], [25], models [1], tools [18] and methods for improved requirements communication [8], rather than investigating what factors cause weak communication of requirements and what effects this has on the final software. To address this gap we report on a case study conducted in the context of large-scale market-driven software development with the following main research questions: (RQ1) what causes gaps that hinder the communication of requirements? and (RQ2) what are the effects of these gaps?

We have performed an explanatory case study at a large market-driven software development company, where we have interviewed nine practitioners. We found a number of communication gaps that affect requirements, mainly in the communication to and from the requirements engineers, but also between roles within development. Four main factors that cause communication gaps have been identified, namely *scale, temporal aspects, common views* and *decision structures*. In addition, nine effects that are a consequence of communication gaps were found, e.g. failure to meet customers' expectations, quality issues and wasted effort.

Section II describes related work. Section III provides a description of the case company. Section IV describes the research method used in this study. Section V contains the results from the interview study, while Section VI describes the outcome of the validation questionnaire on these results. In Section VII we interpret and discuss the results, as well as, limitations of the study. Section VIII contains conclusions and further work.

II. RELATED WORK

Curtis et al studied the upstream part of software development [6] and found that communication between customers, requirements engineers and the development teams is a crucial part in enabling stable requirements and a correct understanding of them, but that for large systems organizational boundaries hinder the communication. It was also found that the communication need is not reduced by documentation [6]. Since communication and interaction with other people is a vital part of requirements engineering (apart from technical skills) soft skills are required to be successful. Based on literature and experience, a classification of such soft skills per requirements engineering activity (e.g. elicitation) has been proposed [19].

978-1-4577-0924-1/11/$26.00 © 2011 IEEE

Communication has also been reported as challenging for distributed software projects that operate in a Global Software Engineering context [25], [27], as it can impede the understanding of requirements, and possibly lead to delays and project failures. Stapel et al. [25] found that many problems for global development can be related to communication, and consist of missing context for interpreting requirements, awareness or documented information. Holmstrom et al. [9] mention temporal distance as challenging in everyday communication in global software development context. Furthermore, even in global software development projects where agile practices were used communication has also been reported as challenging [27]. On the other hand, Kotlarsky and Oshri reported that challenges involved in sharing knowledge across globally distributed teams are still widespread [11]. Finally, Piri reports that many of the common problems encountered in software development projects can be traced back to social factors of the project with special challenges to communicate among distributed teams [20].

Al-Ani and Edwards investigated communication models adopted in large-scale software engineering projects [1]. Others, such as Lutz, investigated linguistic challenges in a global software engineering context [14], while Niinimaki et al. report on findings on communication tools in twelve distributed software projects [18]. The communication flow between different development teams [16] and teams located at different sites [15], [16] have been investigated. The interactions between individuals with different roles in cross-functional development teams have been studied and the majority of missing communication edges was found between people performing roles that were not supposed to be communicating according to the formal organizational structure [16]. For communication around changes that affect multiple development teams it has been reported that there are a handful of key people (called information brokers) [15] that can both facilitate and enable efficient requirements communication, as well as, hinder and/or introduce noise, i.e. misconceptions or erroneous requirements into the requirements communication process.

III. The Case Company

Our results are based on empirical data from industrial projects at a large company that is using a product line approach [21]. The company operates in a market-driven requirements-engineering [10] context that can be characterized by lack of actual customers that can agree to requirements and the continuous inflow of requirements from multiple channels. The company has around 5000 employees and develops embedded systems for a global market. There are several consecutive releases of a platform (a common code base of the product line) where each of them is the basis for one or more products that reuse the platform's functionality. A major platform release has a lead time of approximately two years and is focused on functionality growth and quality enhancements for a product portfolio. For such projects, typically around 60-80 new features are added, for which approximately 700-1000 system requirements are produced. These are then implemented by 20-25 development teams with around 40-80 developers per team, assigned to different projects. The requirements legacy database amounts to a very complex and large set of requirements at various abstraction levels in the order of magnitude of 20,000 entities, making it an example of the Very-Large Scale Requirements Engineering context [22].

A number of different organizational units within the company are involved in the development. For this study, the relevant units are the *Requirements Unit* that is responsible for scope planning and requirements management, the *Software Unit* that develops the software for the platform and the *Product Unit* that develops products based on the platform. Within each unit there are several groups of specialists for different technical areas that are responsible for the work in various stages of the development process. For this case, the most essential groups are the *Requirements Teams (RTs)* (part of the Requirements Unit) that elicit and specify system requirements for a specific technical area, and *Design Teams (DTs)* (part of the Software Unit) that design, develop and maintain software. Each RT has a team leader who manages the team. Another role belonging to the Requirements Unit is the *Requirements Architect* who manages the scope at the high level and also coordinates the RTs. In the DTs there are several different roles, namely

- *Design Team Leader* who leads and plans the team's work for the implementation and maintenance phase
- *Design Team Requirements Coordinator* who leads the teams during the requirements management and design phase, and coordinates the requirements with the RTs
- *Developer* designs, develops and maintains the software
- *Tester* verifies the software

The software unit has a project management team consisting of among others *Quality Managers* who set the target quality levels and *Software Project Managers* that monitor and coordinate the DTs and interact with the Requirements Architects. The product unit is responsible for of the products, for this study *System Testing* is relevant.

The company uses a stage-gate model with several increments. There are *Milestones (MSs)* and *Tollgates (TGs)* for controlling and monitoring the project progress. In particular, there are four milestones for the requirements management and design before implementation starts: MS1, MS2, MS3, and MS4, and three milestones for implementation and maintenance: MS5, MS6, MS7. For each of these milestones, the project scope is updated and baselined. The milestone criteria are as follows:

MS1: At the beginning of each project, long-term RT roadmap documents are extracted to formulate a set of features for an upcoming platform project. A *feature* in this case is a concept of grouping requirements that constitute a new functional enhancement to the platform. At this stage, the features usually contain a description, their market value and effort estimates. The features are reviewed, prioritized

and approved. The initial scope is decided and baselined per RT, guided by a project directive and based on initial resource estimates from the relevant DT. The scope is then maintained and regularly updated each week at a meeting of the *Change Control Board (CCB).* The role of the CCB is to decide upon adding or removing features.

MS2: Features are refined into requirements by the RTs. One feature usually consists of ten or more requirements which are expressed in domain-specific, natural language including many special terms that require contextual knowledge to be understood. Each feature is assigned to a main DT that is responsible for its design, implementation and effort estimates. The requirements for a feature are reviewed together with its main DT and approved.

MS3: DTs refine system requirements and start designing the system. The effort estimates are refined, and the scope is updated and baselined.

MS4: The requirements refinement work and the system design are finished, and implementation plans are made. The final scope is decided and agreed with the software unit.

MS5: All requirements are developed and delivered.

MS6: The software is stabilized prior to customer testing.

MS7: Customer-reported issues are handled. The software is updated and ready to be released.

IV. Research Method

The research was conducted using a qualitative research approach, which is appropriate when individual perceptions of a complex phenomena in its context is to be studied, using a series of interviews [23]. The results reported in this paper are part of a larger study that contains five different RE challenges: 1) Communication gaps, 2) Overscoping, 3) Keeping SRS updated, 4) Monitoring development work from requirements perspective, 5) Manual selection of requirements for release/product. Partial results for challenge 2) Overscoping, were published as a workshop publication [3]. In this paper, we present the results around challenge 1) Communication Gaps. The study has been conducted in three stages, outlined in the sections that follow.

A. Phase one: Pre-study investigation & preparations

In order to seek an explanation and more insight into the challenges around communication of requirements, we selected to perform an explanatory case study [23] where we start by focusing on a specific case. For this approach, we used the experience of one of the authors (from working with requirements, development and processes at the case company) as input in identifying a number of assumed requirements engineering challenges in industry (of which Communication gaps was one), as well as, possible causes and effects of these challenges. In order to avoid selecting a set of assumptions biased by only one person, these assumptions have been iterated upon in a series of brainstorming session with the other authors, and the outcome used as the main input when creating the interview study instrument (which can be accessed online [28]). The following assumed causes of communication gaps were identified in this phase (code within parenthesis denotes the cause to which it is classified in the compiled result, see Section V.A):

- Complex product & large organisation (C1)
- Low understanding of the roles of others (C2)
- Low involvement by RT after req definition (C3)
- Low involvement by DT during req definition (C3)
- Overlapping requirements processes (C3)

B. Phase two: Interview study at the case company

To facilitate the discussion regarding requirements communication, and support exploring and enriching the understanding of this complex phenomenon, the qualitative interview study method has been utilized. The interview instrument [28] produced in phase I (see Section IV.A) was designed to be semi-structured with a high degree of discussion between the interviewer and the interviewee. For each of the main challenges (including communication gaps) an open ended question about the challenge was asked: if it was a challenge, what causes it and what effects it has. This was done to find the causes and root causes of the main challenges without imposing the assumptions made during the pre-study on the interviewee. If the interviewee did not explicitly mention an assumed cause they were specifically asked about their view on it. The resulting theory related to communication challenges has thus been grounded in the empirical data gathered from interviewee with minimized bias from researchers [26].

TABLE I. Interviewees: code (First letter denotes organizational belonging), unit and Role(s) (See Section III)

Code	Organizational unit	Role (experience in years)
Ra	Requirements	RT leader (5 years)
Rb	Requirements	RT leader (2 years)
Rc	Requirements	Requirements architect (3 years)
Pd	Product	System test manager (7 years)
Se	Software	Tester (3 years)
Sf	Software	Software project manager (2 y), DT leader (2 y), Developer (2 y)
Sg	Software	Quality manager (3 years)
Sh	Software	DT requirements coordinator (0,5 y), Developer (2 y), DT leader (1 year)
Si	Software	DT requirements coordinator (7 years)

The interviews were scheduled for 90 minutes each with the possibility to reduce time or prolong it. All interviews were recorded and transcribed, and the transcripts sent back to the interviewees for validation. The coding and analysis was done in an integrated and iterative fashion. The underlying structure of the interview instrument was used for categorizing the views of the interviewees. For each interview, the transcribed chunks of text were placed within the relevant sections and, if so needed, copied to multiple sections. The used sections, or categories, correspond to the challenges, causes and effects (both assumed and mentioned during the interviews.) These were numbered to facilitate consolidating between the interviews. Relationships were

captured by noting dependencies to and from each category in specific columns.

In order to cover the full project life cycle from requirements definition through development to the end product people from all relevant organizational units (Requirements, Software and Product, see Section III) were selected. Nine persons were selected (by the researchers) to be interviewed. Two of the interviewees (with identical roles) requested to have their interview together. The roles, organizational belongings, and length of experience for each interviewee can be found in TABLE I.

C. Phase Three: Validation of results with practitioners

In the third phase of the case study, the results from the interviews were presented to (another) seven practitioners who were asked to state their view on the results of the study via a questionnaire (see Section VI). The following practitioners were selected (by the researchers): four people from the software unit (a Software project manager and, from the Development teams, a team leader, a requirements coordinator, and a tester), 2 people from the requirements unit (Requirements team leader and Requirements architect) and one person from the product unit (System test manager). These 7 practitioners have worked within the company for a range of 4 to 13 years. At a meeting, the results around communication gaps (see Section V) were presented, briefly discussed (especially around disagreements and additional viewpoints not covered in the results), and the participants filled out a questionnaire (available online [28]) stating to which degree they agree to the results, and if they see additional, causes, root causes, and effects for communication gaps and connections to other RE challenges. The session was scheduled for 90 minutes with the possibility to extend or decrease the time as needed. Due to scheduling difficulties two sessions were required to cover all participants.

V. RESULTS

The results of the interview study are divided into four parts. Section V.A covers the causes of communication gaps, Section V.B contains the root causes of the main causes, Section V.C describes the effects of communication gaps, and Section V.D covers the connections found between communication gaps and the other challenges covered by the study. The results of the questionnaire (study phase three, see Section IV.C) are reported in Section VI.

A. Causes of Communication Gaps

While analyzing the results, we identified three of the assumed causes (see Section IV.A) as exhibiting a temporal aspect, i.e. some roles are available at different times and phases throughout the lifecycle. These assumed causes were grouped into the (new) cause *Gaps between roles over time* (C3). In addition, a fourth main cause was identified based on three of the eight interviewees mentioning issues related to company-wide strategy and unclear business priority of scope, which affects the requirements communication. The cause *Unclear vision of overall goal (C4)* was added to cover this. For each of the causes, the interviewees' viewpoints were categorised per organization. The results are presented in TABLE II. , using the following classification:

Experienced: cause (occurrence and impact on challenge) is experienced and was mentioned without prompting

Agreed: cause not directly mentioned, but derived, agreed to direct question, observed or heard from others

Partly agreed: partly *Experienced* or partly *Agreed*

Disagreed: does not agree that this causes the challenge

Not mentioned: not within expected experience for role

All of the nine interviewees had *Experienced, Agreed* or *Partly Agreed* to communication gaps being a challenge, and a majority of the interviewees have *Experienced* or *Agreed* to causes 1 (9 of 9) and 2 (5 of 9) contributing to gaps in communication of requirements.

TABLE II. NO OF INTERVIEWEES WHO MENTIONED EACH CAUSE OF COMMUNICATION GAPS PER ORGANIZATIONAL UNIT (R=REQUIREMENTS, S=SOFTWARE, P=PRODUCT)

	Communi-cation gaps			C1 Complx product & large org			C2 Low understand-ing of roles			C3 Gaps bt roles over time			C4 Unclear vision of goal		
Organizationl unit	*R*	*S*	*P*	*R*	*S*	*P*	*R*	*S*	*P*	*R*	*S*	*P*	*R*	*S*	*P*
Experienced	2	2	1	3	4	1	1	2	1		2	1	2	1	
Agreed	1				1		1			1					1
Partly agreed		3					1	2		2	2				
Disagreed													1		
Not mentioned								1			1			4	

Communication gaps (as a challenge) Three of the interviewees (Sg, Sh, Si) *Partly Agreed*, with the motivation that the communication gaps vary between teams; for some there is close communication, for others the requirements are not communication to the affected people.

Complex product & large organization (C1) All interviewees mentioned that size impacts both agreeing on requirements and communicating them to others. For example, Rc said 'There are many people who need to be involved and have an opinion on things.' While Sh said: 'No-one knows the full extent of what the product can do, not even within the company.' Interviewee Rb believes that the organizational structure has a huge impact on the communication and the result of development projects.

Low understanding of roles of others (C2) Sh and Si (*Partly Agreed)* both mentioned that the understanding of requirements-related roles is weak within the development teams, with the exception of the DT requirements coordinator. The DT tester (Se) *Experienced* weak understanding of testers potential to contribute to requirements work, e.g. ensuring verifiability. Interviewee Pd *Experienced* lack of consideration of system aspects by the RTs and DTs due to a weak understanding of the role of system test. Rc (*Partly agreed)* stated that communication between RT and DT teams improved with increased understanding of each other's roles.

Gaps between roles over time (C3) One of the RT leaders (Ra) *Agreed* to this cause, and has experienced that direct communication with the DT throughout the life cycle (i.e. no

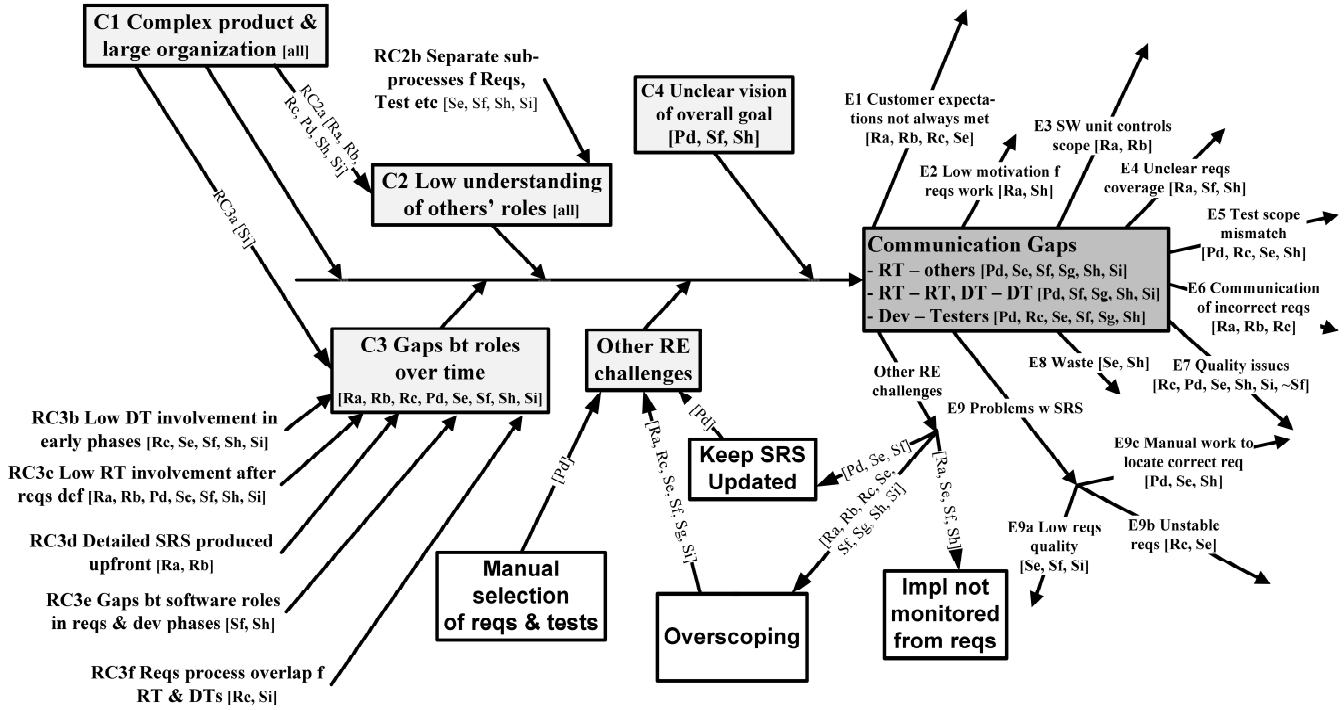

Figure 1 Causes (C), root causes (RC) and effects (E) of communication gaps, interviewee code within brackets.

gaps in time) results in more insight into and control of what is implemented. Four of the interviewees *Partly agreed* to this cause; Ra, Rc and Sh mentioned both, periods in time when requirements communication between RT and DT was sufficient (e.g. Requirements architect continuously involved via change management process), and periods when it was not so (e.g. lack of tester involvement in early phases.) Si mentioned that the Requirements Teams do not always provide requirements in a timely fashion.

Unclear vision of overall goal (C4) Both RT leaders (Ra and Rb) expressed a lack of clear vision and strategies that can be used in a practical way when defining requirements for new products. This leads to power struggles between different units and technical areas rather than constructive communication around how to reach a common goal. Pd *Agreed* to this and described that there is a lack of communication around quality and system-level requirements. In contrast, interviewee Rc *Disagreed* to this cause since the technical roadmaps are reviewed and aligned with company strategy early in the projects.

B. Root Cause Analysis

To provide a deeper understanding around the causes of communication gaps, the interviewees were asked to describe the root causes that may be triggering these gaps for each cause. The assumed causes that were categorized as C3 (see IV.A) are included as root causes of C3. Figure 1 summarizes the full picture of our interpretation of the interview material including the root causes (denoted RC.)

Root causes of C1 Complex product & large organization is the nature of the case company and its products, and its root causes are out of scope for this study.

Root causes of C2 Low understanding of roles of others The complexity of the products (RC2a) requires many skills that are spread over many different roles. The interviewees describe that it is hard to get an understanding of the big picture concerning how they should work and the purpose and responsibility of different roles, both due to the sheer numbers of roles involved (RC2a), as well as, the way the process is described in separate sub-processes for each discipline (RC2b), e.g. requirements and test. This affects the communication around requirements, causing gaps when people do not know or understand the roles of others, e.g. the difference in work characteristics between the RT leaders (standardisation & requirements work) and the DTs (design, development & maintenance.)

Root causes of C3: Gaps between roles over time The work is distributed over many different people and roles (RC3a), which vary over the life cycle of a project. Our interviewees clearly describe that it is hard to achieve continuity over time especially at the handover points when work is passed on to new roles. The time periods mentioned for such gaps are, from initial scope selection to requirements detailing (RC3b), i.e. MS0-MS2, through the design and planning phase (RC3d), i.e. MS2-MS4, and then in the implementation, testing and later phases (RC3c, RC3e), i.e. MS4-. During all of these phases there is a need for requirements communication between RTs and DTs, but (as our interviewees describe) the level of communication varies between the teams. Ra expressed the situation in this

way: ’We deliver requirements, but if you aren't actively checking all the time that they [DT] are implementing according to the requirements, it is quite often the case that it is something different that is being implemented.’ An even more critical point in time (described by Sf and Sh) is at MS4 when the implementation work starts, and the DT and software project responsibility is handed over to new roles without awareness of requirements (RC3e). This handover results in the requirements being more or less ignored after MS4.

Root causes of C4: Unclear vision of overall goal Vision and strategy guidance are expected to be provided by the company management. The root causes for lack of this are out of the scope of the study.

C. Effects of Communication Gaps

E1 Customer expectations not met When working with customer-specific requirements, communication is even harder, which Se expressed as ‘Just getting the right specification [from the customer] was impossible. And then when we finally got it, it was outdated and there was a new one.’ In addition, it is not unusual that customers are promised features that are not agreed to by the software unit, which may result in failure to meet them.

E2 Low motivation to contribute to reqs work The communication gaps around requirements between RT and DT were mentioned as leading to decreased motivation among RT leaders to work with requirements. Low understanding of roles leads to some DT testers not seeing any value in participating in requirements work.

E3 Software unit controls what is implemented Due to communication gaps between the Requirements Unit and the Software Unit, the Software Unit (with development resources) control what is finally implemented. In addition, the Software Unit has an internal roadmap that covers more than architectural improvements, and which is not agreed with the Requirements Unit.

E4 Unclear requirements coverage One of the RT leaders (Ra) said that if he does not stay in touch with the DT, he never knows exactly what is implemented. The communication gaps caused by C2 and C3 lead to DTs neither discussing requirements problems with the RTs (e.g. unclarities), nor informing them of changes that affect requirements.

E5 Test scope mismatch The test scope executed by system test is based on the SRS, but since the SRS does not correctly reflect the requirements that are finally implemented (see E4) a lot error reports are created on functionality that is not designed to work according to the SRS. Pd said: ’If you look at the error reports that are submitted, the number of things that are rejected [by DTs] due to being intended to be like this, increases in the later phases because you [system testers] are [physically] further away from requirements and developers.’ The communication gaps between RTs and DTs and System test are causing testers to verify invalid requirements for which the changes have not been communicated.

E6 Communication of incorrect reqs When requirements frequently change (which they do in a market-driven context) and also slip through the gaps, it is very hard to communicate correct requirements, both to the customers and internally. Ra said: ’We gave them [customers] information about what we thoughts would be included, which often was completely wrong.’

E7 Quality issues The lack of direct communication between RTs and testers, both system testers and DT testers, lead to weak focus on system aspects (e.g. quality requirements), testing requirements (e.g. test harnesses) and test cost, in early project phases, resulting in quality issues later on. In contrast, Sf stated that gaps between developers and testers are beneficial for software quality, since competition encourages testers to smoke out problems with software produced by the developers.

E8 Wasted effort The communication gaps increase the time it takes to communication changes to all involved parties, and thus increase the amount of work wasted so far on requirements, design and implementation work, which then has to be redone. The gaps caused by roles changing at MS4 leads to waste of effort to transfer knowledge, and missed requirements knowledge and awareness.

E9 Problems with SRS The gaps between RT leaders and, DT testers and developers, result in unclear, ambiguous and non-verifiable SRS requirements (E9a), and subsequent problems when implementing and verifying them. The communication gaps between RT and DT during requirements detailing contribute to unstable requirements (E9b); since the viewpoints of the testers and the developers are not taken into consideration until later project phases. Instead, issues are uncovered when design, implementation and testing start at which point the requirements need modifying. The problem is enhanced when external parties like customers are involved. The communication gaps between RTs and developers and testers result in them being force to locate requirement information (E9c) mainly through other channels. The SRS is one such channel, in which it is hard to locate specific and relevant requirements and sometimes the implemented requirements are not in the SRS (see E4). The DT testers mainly receive requirements by asking the developers.

D. Connections to Other Challenges

When analyzing the interviews, we found that of the four other challenges covered by the study, all of them had connections to communication gaps, either mentioned as direct causes or consequences of communication gaps, or by resulting in an effect that contributes to another challenge. The full picture of the connections is shown in Figure 1.

Overscoping, or including more requirement than there are resources for, results in increased communication gaps between teams (both DTs and RTs), because they do not have time to communication around requirements. Overscoping also results in friction between the DTs and software project managers, e.g. when failing to deliver according to plan.

Gaps in communication between the RTs and stakeholders, as well as, DTs, lead to the RTs specifying a scope missing vital requirements and without reliable cost estimates, all of which leads to overscoping.

Keeping SRS updated partly bridges the gaps in communication between RT and system test. But, when the SRS is not kept updated, this results in error reports on invalid SRS requirements (see E5 in Section V.C) and increased communication gaps to DTs who claim that the software works as it should.

Communication gaps between RTs and DTs in later project phases result in RTs being unaware of implementation changes that affect the requirements, causing a mismatch between SRS and delivered software.

Manual selection of reqs for products contributes to communication gaps for the same reasons as for the challenge *Keep SRS updated* since the product requirements are selected from the requirements in the SRS, which is not in line with the implemented software (see E4 and E9 in Section V.C)

Implementation not monitored from reqs viewpoint is caused by **g**aps between roles before and after MS4 (see RC3e in Figure 1.) When implementation **s**tarts, the responsibility is transferred to roles who have little insight or awareness of requirements, during project phases when RTs have little contact with DTs. Requirement change, but often without RT involvement. The implementation continues, more or less, without being concerned with the requirements.

VI. Validation of Results with Practictioners

In phase three of the study (see Section IV.C), the results described in Section V were presented to seven practitioners at the case company. They noted their level of agreement in a questionnaire [28] using the following notation:

Experienced: I have experienced this to be valid
Agree: I agree to this, but have no personal experience
Partly agree: I agree to part, but not all, of this
Disagree: I do not agree
Don't know: I have no knowledge of this

A majority of the participants noted *Experienced* or *Agreed* to all, but one, of the causes, root causes, and effects. For *Overlapping requirements process between RT & DT* (RC3f, see Figure 1), three respondents had *Experienced* this root cause, while three *Partly agreed* and one answered *Don't know*. In addition to the presented results, late test involvement in the projects was mentioned as an additional root cause to C3 *Gaps between roles over time*, resulting in missing requirements from the testers concerning, e.g. test harnesses and other functionality required for verifying the software. Concerning C1 *Complex product & large organization*, one participant claimed that the way the product portfolio was planned (by business people with little input from the software unit) resulted in a more complex portfolio than necessary since little consideration was given to the cost of implementing and supporting a large number of different configurations.

VII. Interpretation and Discussion

In this section, we provide our interpretation and discussion of the results around causes and effects of communication gaps, and compare them to related work. In Section VII.A, we discuss the limitations of this study.

Requirements communication is a challenge for the case company though there are examples of good requirements communication between teams and individuals. The four identified causes correspond to four different factors that contribute to communication gaps, namely *scale (C1), common views* (C2), *temporal aspects* (C3) and *decision structures* (C4).

Cause 1: Complex Product and Large Organization covers the factor of *scale.* Our responders clearly state that the size of the organization and the complexity of the products, contribute to communication gaps. A survey study into coordination of large-scale software development [12] found that scale contributes to communication gaps over geographic, organizational and social boundaries, due to dividing the work over many different specialized roles. In addition, organizational boundaries cause communication gaps that hinder the mutual understanding of requirements [6]. Our study shows that there is a communication gap upstream towards the Requirements Teams resulting in requirements being received by Development Teams from many different sources, as well as, incomplete requirements specifications, overscoping, and conflicting requirements.

Cause 2: Low understanding of each other's roles covers the factor of *common views*. The domain knowledge and perspectives vary between roles. Without respect and mutual understanding for each other's viewpoints this causes communication gaps, either by not communicating at all (due to lack of understanding that other roles are impacted) or by ineffective communication (e.g. missing tacit requirements due to lack of insight into the customer's domain.) Weak understanding of the work of other units negatively affects the communication and cooperation [24]. Communication around the design between stakeholder and architects leads to shared understanding of the requirements and identification of tacit requirements, as well as, needed requirement changes [8]. Similarly, application domain knowledge has been reported as vital in designing a solution that will meet the customer's needs [6].

Cause 3: Gaps between roles over time covers the factor of *temporal aspect*. Our results indicate that requirements communication needs to continue throughout the project life cycle, since requirements are dynamic and change, often until they are implemented. Communication gaps between requirements and development teams during early phases have previously been found to result in requirements that could not be implemented [6], [2], [24]. Failure to bridge these gaps results in delays, and increases the cost of handling late errors and changes [2]. Also, there are certain hand-over points (MS2 and MS4 for our case company) when it is crucial that sufficient knowledge of the requirements is transferred to new roles, in order to ensure continuity throughout the project life cycle and avoid development becoming disconnected from requirements. A suggestion for how to avoid some of these gaps is given by Fricker et al. [8], were communication between stakeholders and architects around design was shown to improve the probability that the requirements are carried on into later phases of the project. A surprising detail of our results indicate that producing a detailed requirement specification upfront may contribute to communication gaps (root cause RC3d), since it then may be assumed that no additional communication of requirements is needed. We found similar conclusions drawn by Curtis et al. [6], i.e. that the existence of artefacts can contribute to communication gaps since people tend to assume the artefacts in themselves constitute sufficient communication.

Cause 4: Weak vision of overall goal covers the factor of *decision structures*. When there is no clear common goal for the software development it is up to the individual teams and units to make decisions on which requirements to include. For our case company, this, in combination with weak understanding of each other's roles (C2) has led to wide communication gaps between the Requirements and the Software Units, resulting in the Software Unit controlling which requirements are actually implemented (E3). Similar communication gaps are reported by Karlsson et al. [10] as a challenge for which having a common goal and vision (C4) is a way to resolve, or close, such gaps.

Effects Communication gaps contribute to a number of consequences for the project and for the resulting software. The communication gaps during requirements definition contribute to an instable, unclear and ambiguous SRS (E9.) Weak communication with the customers has been found to cause instable requirements [6], while communication between the customer and the development team is seen to mature both the requirements and the design. For our case company (that operates in a consumer market with no direct communication with the end customers) the Requirements Unit represents the (anonymous) customers. The view of what constitutes a *good* requirements has been found to vary between roles [10], indicating a weak common view (C2).

For the case company, there is a huge gap in requirements communication during the later phases of the projects (after MS4), which results in the software implementation being done without the projects, or teams, being monitored from a requirements perspective. Instead, project management monitors on committed delivery dates and number of error reports, while the developers rely on the design correctly reflecting the requirements, and the testers rely on the SRS being kept updated (which is a challenge.) In large-scale market-driven development where change is constant, this results in unclear requirements coverage (E4); there is no clear and common view of which requirements that are actually supported. Instead, incorrect requirements information is given (E6), both internally and to customers, also mentioned as a consequence of weak communication [10], and the test scope does not match the implemented requirements (E5). All this results in not always meeting the customers' expectations (E1); either due to lack of desired functionality or quality issues (E7), also reported by Flemming [7]. In addition, effort is wasted (E8), e.g. when testing requirements for which agreed changes have not been communicated, which contributes, together with C2, to low motivation to work with requirements (E2).

A. Threats to Validity and Limitations

We discuss the validity threats according to the classification provided by Robson [23]. The main threat to *description validity* is to provide a valid description of what interviewees said and meant. This threat was addressed by recording and transcribing the interviews. The transcripts were sent back to the interviewees to check for misinterpretations and other errors. To ensure open and honest replies the interviewees had full anonymity; the full set of names of the interviewees was only know to the researchers and the company is large enough for the individuals not be identifiable from the information given about them in this paper.

To address the treats to *valid interpretation*, the question on each challenge (of which communication gaps was one) were formulated in an open and indirect way to encourage the interviewee to express her own opinion before mentioning the assumed causes. A possible source of unreliability is related to *observer biases* where the results from the pre-study, as well as, questions asked during the interview, may have been consciously or unconsciously biased by the researcher. This threat was addressed by all the authors discussing the results of the pre-study, the selection of interviewees, and reviewing the interview instrument. Moreover, the practitioner's involvement in the study has played a vital role in focusing on and ensuring that the problems under investigation are authentic problems, that the interpretation of data is based on a deep understanding of the case and its context, and that the outcome of the study is authentic. To mitigate the risk of quotations becoming out of context during the analysis phase [5], the observer triangulation method was used [23]; one researcher randomly selected two interview recordings and performed an independent transcription and coding.

Differences were discussed and conflicts resolved. Data triangulation was also applied by the questionnaire responses from another set of practitioners to further validate the results from the interview study.

The possibility of generalizing the results of this case study has been addressed both internally within the study and in respect to external generalisability. The *internal generalisability* was addressed by sampling participants from different parts of the company with different roles. As for *external generalisability,* the main threat to validity is no possibility of performing a statistical generalization due to lack of representative sample and only one company involved in the study. However, the main focus on this study is to increase the understanding of communication around requirements and explore possible causes of gaps in this communication rather than providing a full theory that can be generally applied. Finally, communication gaps were confirmed as a challenge by all our responders with only minor differences of the importance of this issue and all of the identified causes, and several of the effects, of communication gaps have been reported by other researchers in related studies (see Section VII.)

VIII. Conclusions and Future Work

Communication is one of the key mechanisms in coordinating a project, of which the requirements, or 'a common view of what the software they are developing should do' [12], is a vital part. The organizational theory literature suggests that for an organization to be successful, an appropriate combination of organizational structure, processes, and communication and coordination mechanisms, is needed [4]. Since software development is as highly collaborative endeavour, many of the problems encountered during software projects can be traced back to social factors [20]. Despite the fact that several studies have reported the challenging nature of communication in software and requirements engineering [7], [9], [10], [11], [13], [20], [27] and investigated various aspects of communication [1], [14], [15], [16], [18], no consolidated empirical evidence on the causes, root causes effect and relations to other requirements engineering challenges has (to the best of our knowledge) been presented.

In this paper, we address this gap by reporting empirical evidence based on an interview study performed with nine interviewees at a large software development company. To further strengthen the validity of the study we also conducted a questionnaire with a different set of seven practitioners who confirmed the results. The study confirms that communication is a challenging part of requirements engineering and may cause a situation where requirements *slip through the gaps*; are misinterpreted or overlooked, resulting in failure to meet customers' expectations both concerning functionality, as well as, quality.

We have identified four main factors that may cause communication gaps: *scale*, *common views*, *temporal aspects*, and *decision structures*. The size and complexity of the software development, i.e. *scale*, increases the challenge of requirements communication. We found communication gaps between the requirements engineers and a number of stakeholders, resulting in missing requirements, e.g. for quality. Instead, these requirements surface in later phases, thus, incurring increased cost. *Common views* and mutual understanding are necessary for communication to be productive. Weak understanding of each other's roles and responsibilities causes gaps in communication. For example, the testers' competences are not utilized when defining and reviewing requirements, or the requirements engineers are not consulted when making implementation choices that affect the requirements. *Temporal aspects* come into play when there is a lack of continuity in requirements awareness through the project life cycle. This may cause gaps in the requirements communication. Hand-over points, e.g. defined by the process, where the responsibility is passed on to new roles constitute a risk of missing vital requirements knowledge and awareness. This may result in requirements being misunderstood and incorrectly implemented, or, making decisions that affect the requirements without considering all relevant aspects. For example, if there is no requirements awareness in the implementation phase, the developers tend to make their own requirement modifications without considering the impact on the customer or on other parts of the development organization, such as test. *Decision structures* also contribute to communication gaps. Weak, or unclear, visions or goals for the software development (due to not being communicated or not being clear enough) contributes to weak communication, primarily, between those defining the requirements and the development unit, since there is no mutual understanding of the goal.

Our study shows that communication gaps can have serious and expensive consequences in terms of wasted effort and quality issues, as well as, not meeting the customers' expectations and even communicating an incorrect picture of what requirements a product fulfils to the customers. In addition, communication gaps can contribute to a number of other RE-related challenges, like overscoping and keeping the SRS updated. This, in turn, contributes to communication gaps, i.e. the software development ends up in a vicious cycle.

The increased understanding of the causes and risk of gaps in requirements communication provided through this study, can be a help in identifying potential communication gaps in existing software development processes and organizations. The goal should be to close such gaps and enable requirements management to efficiently support and guide development projects towards producing quality software that will meet customers' expectations.

Future work includes investigating how aspects such as organizational set-up, software development model (agile or waterfall) and application of different software engineering methods affect the challenges, and their causes both within the case company, and in a broader context.

ACKNOWLEDGEMENT

We would like to thank all anonymous interviewees and questionnaire respondents for their contribution to this project. The project is partly funded by the Swedish Foundation for Strategic Research and VINNOVA within the EASE and UPITER projects.

REFERENCES

[1] B. Al-Ani, H. K. Edwards, "A Comparative Empirical Study of Communication in Distributed and Collocated Development Teams." Proc. IEEE Int. Conference on Global Software Engineering (ICGSE '08). IEEE Computer Society, pp. 35-44. doi: 10.1109/ICGSE.2008.9.

[2] D. M. Berry, K. Czarnecki, M. Antkiewicz, M. AbdElRazik, "Requirements Determination is Unstoppable: An Experience Report" Proc. IEEE Requirements Engineering Conference (RE'10), IEEE Computer Society, Sept 2010, pp. 311, doi: 10.1109/RE.2010.44

[3] E. Bjarnason, K. Wnuk, B. Regnell, "Overscoping: Reasons and Consequences – A Case Study in Decision Making in Software Product Management", Proc IEEE Int. Workshop on Software Product Management (IWSPM'10), IEEE Press, Sept 2010, pp. 30-39, doi: 10.1109/IWSPM.2010.5623866

[4] M. Cataldo, M. Bass, J. D. Herbsleb, and L. Bass. "On Coordination Mechanisms in Global Software Development," Proc IEEE Int Conf. on Global Software Engineering (ICGSE '07). IEEE Press, pp. 71-80, doi: 10.1109/ICGSE.2007.33

[5] A. J. Coffey and P. A. Atkinson, "Making Sense of Qualitative Data: Complementary Research Strategies", Sage Publications, Inc, 1996.

[6] B. Curtis, H. Krasner, and N. Iscoe, "A field study of the software design process for large systems." *Commun. ACM,* vol. Nov. 1988, pp. 1268-1287, doi:10.1145/50087.50089.

[7] W. R. Flemming, "Requirements communication", Int. Conf. Automatic Testing (AUTOMTESTCON'78), pp. 228-229, 1978, doi: 10.1109/AUTEST.1978.764370

[8] S. Fricker and M. Glinz, "Comparison of Requirements Hand-Off, Analysis, and Negotiation: Case Study", Proc IEEE Int. Requirements Engineering Conference, Sept 2010, pp. 167-176, doi: 10.1109/RE.2010.29

[9] H. Holmstrom, E. O. Conchuir, P. J. Agerfalk, and B. Fitzgerald, "Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance." In Proc IEEE Int Conf. on Global Software Engineering (ICGSE '06), IEEE Computer Society Press, pp. 3-11, doi: 10.1109/ICGSE.2006.261210

[10] L. Karlsson, Å. G. Dahlstedt, J. Natt Och Dag, B. Regnell, and A. Persson, "Requirements engineering challenges in market-driven software development An interview study with practitioners", Inf and Soft Techn, vol. 49, Dec. 2007, pp. 588-604, doi: 10.1016/j.infsof.2007.02.008.

[11] J. Kotlarsky and I. Oshri. "Social ties, knowledge sharing and successful collaboration in globally distributed system development projects." Eur Journal of Inf Systems, vol. 14, Mar. 2005, pp. 37-48, doi: 10.1057/palgrave.ejis.3000520.

[12] R.E. Kraut and L. Streeter, "Coordination in Software Development", Communications of the ACM, vol. 38, Mar 1995, pp. 69-81, doi: 10.1145/203330.203345.

[13] M. Lubars, C. Potts and C. Richter, "A Review of the State of the Practice in Requirements Modeling" Proc. IEEE Int. Symposium on Requirements Engineering (RE'93), IEEE Press, Jan. 1993, pp. 2-14, doi: 10.1109/ISRE.1993.324842.

[14] B. Lutz, "Linguistic Challenges in Global Software Development: Lessons Learned in an Int. SW Development Division,", Proc. Int. Conf. on Global Software Engineering, Jul. 2009, pp. 249-253, doi: 10.1109/ICGSE.2009.33

[15] S. Marczak, D. Damian, U. Stege and A. Schröter, "Information Brokers in Requirement-Dependency Social Networks", Proc. IEEE Int Conference on Requirements Engineering, IEEE Press, Sep. 2008, pp. 53-62, doi: 10.1109/RE.2008.26.

[16] S. Marczak, I. Kwan and D. Damian, "Investigating Collaboration Driven by Requirements in Cross-Functional Software Teams", Proc. Int. Workshop on Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills, Aug. 2009, pp.15-22, doi: 10.1109/CIRCUS.2009.2.

[17] M. D. Myers and D. Avison, "Qualitative Research in Information Systems", Sage Publications, USA, 2002.

[18] T. Niinimaki, A. Piri, C. Lassenius and M. Paasivaara, "Reflecting the Choice and Usage of Communication Tools in GSD Projects with Media Synchronicity Theory" Proc. IEEE Int. Conf. on Global Software Engineering, Sep. 2010, pp. 3-12, doi: 10.1109/ICGSE.2010.11

[19] B. Penzenstadler, T. Schlosser and G. Frenzel, "Soft Skills REquired: A practical approach for empowering soft skills in the engineering world", Proc. Int. Workshop of Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills, IEEE Computer Society, Aug. 2009, pp. 31-36, doi: 10.1109/CIRCUS.2009.5.

[20] A. Piri, "Challenges of Globally Distributed Software Development - Analysis of Problems Related to Social Processes and Group Relations", Proc IEEE Int Conf on Global Software Engineering, Sep. 2008,pp. 264-268, doi: 10.1109/ICGSE.2008.33.

[21] C. Pohl, G. Böckle, F. J. van der Linden, "Software Product Line Engineering: Foundations, Principles and Techniques", Springer-Verlag, New York USA, 2005.

[22] B. Regnell, R. Berntsson-Svensson, K. Wnuk, "Can We Beat the Complexity of Very Large-Scale Requirements Engineering?", Proc. Int. Conf. on Requirements Engineering: Foundation for Software Quality (REFSQ '08), vol. 5025 of LNCS, Springer-Verlag, Berlin, Heidelberg, pp. 123-128. doi: 10.1007/978-3-540-69062-7_11

[23] C. Robson. "Real World Research" Blackwell, 2002.

[24] G. Sabaliauskaite, A. Loconsole, E. Engström, M. Unterkalmsteiner, B. Regnell, P. Runeson, T. Gorschek, R. Feldt, "Challenges in Aligning Requirements Engineering and Verification in a Large-Scale Industrial Context", Proc. Int. Working Conference Requirements Engineering: Foundation for Software Quality (REFSQ 2010), Mar. 2010, LNCS, vol. 6182, pp. 109 – 115, doi: 10.1007/978-3-642-14192-8_14.

[25] K. Stapel, E. Knauss, K. Schneider. "Using FLOW to Improve Communication of Requirements in Globally Distributed Software Projects." Proc. Int. Workshop on Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills. IEEE Computer Society, Aug. 2009, pp. 5-14, doi=10.1109/CIRCUS.2009.6.

[26] A. Strauss, J. Corbin, "Basics of Qualitative Research: Grounded Theory Procedures and Techniques", Sage Publications, New York, 1990.

[27] R. Urdangarin, P. Fernandes, A. Avritzer, D. Paulish, "Experiences with Agile Practices in the Global Studio Project" Proc. IEEE Int. Conf. on Global Software Engineering, Aug. 2008, pp. 77-86, doi: 10.1109/ICGSE.2008.11.

[28] The interview instrument and validation questionnaire for the Before aNd After study (BNA) is available at http://serg.cs.lth.se/research/experiment_packages/bna