Please quote as: Hoffmann, A.; Bittner, E. A. C. & Leimeister, J. M. (2013): The Emergence of Mutual and Shared Understanding in the System Development Process. In: Requirements Engineering: Foundation for Software Quality, Lecture Notes in Computer Science. Hrsg./Editors: Doerr, J. & Opdahl, A. L. Verlag/Publisher: Springer Verlag, Essen, Germany. Erscheinungsjahr/Year: 2013. Seiten/Pages: 174-189.

The Emergence of Mutual and Shared Understanding in the System Development Process

Axel Hoffmann[1], Eva Alice Christiane Bittner[1] and Jan Marco Leimeister[1,2]

[1]Information Systems, Kassel University, Pfannkuchstr. 1, 34121 Kassel, Germany
[2]Insitute of Information Management, University of St. Gallen, Mueller-Friedberg-Strasse 8, CH-9000 St.Gallen, Switzerland
{axel.hoffmann, eva.bittner, leimeister}@uni-kassel.de

Abstract. [Context and motivation] In interdisciplinary requirements engineering, stakeholders need to understand how other disciplines think and work (mutual understanding) and agree on the system they develop (shared understanding) in order to collaborate effectively. **[Question/problem]** In this paper we analyse extent and forms of (lacking) mutual understanding according to the periods in the process of conceptual change. **[Principal ideas/results]** We analyse the communication of a multidisciplinary team while developing a mobile application. Although the team tried to resolve differences in meaning early on by applying approaches for clarification, questions for consolidation, exploration and elaboration occurred at different points in time throughout the process. Even when artefacts were already agreed upon, the development team explored lack of mutual understanding to underlying concepts or relationships. A revised shared understanding led to adjustments of the artefacts and thus hampered the process. **[Contribution]** We therefore call for research that explores ways of systematically building mutual and shared understanding in the development process.

Keywords: Mutual Understanding, Shared Understanding, Requirements Engineering, System Development Process

1 Introduction

It is widely acknowledged that mutual and shared understanding between stakeholders is important for successful development projects [1]. This is especially true for the requirement engineering activities [2-4]. Stakeholders need to understand what other stakeholders are able to understand and work with, and they need to deliver artefacts that can be used by others [5]. Further, the stakeholders need to agree on and determine the system that is built in subsequent activities.

When developing socio-technical systems many stakeholders from various backgrounds are involved in requirement engineering activities. This interdisciplinary development enhances the importance of a shared understanding of the system and the requirements. While the stakeholders involved usually do not need to be experts in all fields tackled by the development project, “they have to be able to integrate their

knowledge bases in a sensible manner" [6]. Coming from different disciplines, actors might - without noticing - be using the same words for different concepts or different words for the same concepts [7]. They might be unaware of unshared individual knowledge crucial for completing the task successfully (lack of mutual understanding). Or even if they are aware of differences in knowledge and understanding, they might not agree on a shared perspective at an early stage (lack of shared understanding). This can lead to substantial losses in efficiency in collaboration processes and suboptimal outcomes [8-10]. Necessary late changes to requirements are likely to be followed by evitable rework and time-consuming changes to the whole system. Unfortunately, assessing whether a shared understanding of the system exists is not trivial. Various ideas and views only become evident in the course of the project, making a potential adjustment of the system and its requirements necessary.

As we identify shared understanding as a key success factor of interdisciplinary development projects and as a dynamic state that changes through interaction and communication, we aim to examine the interactive process of building shared understanding throughout a real world software development project. This paper explores in which stages of the development project a lack of mutual or shared understanding is discovered and how this is resolved. Different sources of disagreement require different strategies to resolve them [11], and an understanding of the causes of lacking shared understanding is also necessary. Therefore, we further investigate which different types of conflicts are discovered at which phases. Thus, we show particular types of understanding that should be improved by using additional effort. To achieve this, we examine the evolution of shared understanding on properties and requirements of a mobile application in an interdisciplinary development project by focusing on development artefacts and the correlating communication. We categorise questions and hints that are raised by stakeholders according to the process of conceptual change. Consolidation and exploration questions indicate effort to gain mutual understanding. Elaboration questions try to reconcile different understandings or resolve conflicts.

This paper has been structured in the following manner. First, a short explanation of what mutual and shared understanding will be presented, as well as how they can be achieved in development projects. Subsequently, the research design of the case study will be described, including the team, the development approach, data collection and data analysis. Further, we report and discuss the results. The paper closes with limitations and implications for further research.

2 Mutual and Shared Understanding in Development Projects

We define shared understanding as the ability of multiple agents within a group to coordinate behaviours towards common goals or objectives based on mutual knowledge, beliefs and assumptions on the task, as well as the group, the process or the tools and technologies used that may change throughout the course of the group work process and may impact group work processes and outcomes [12]. This

definition implies a dynamic (process) view of shared understanding. Mohammed et al. [10] note, that "in order for a team to achieve a shared, organized understanding of knowledge about key elements in the relevant environment, changes in the knowledge and/or behavior of team members will most likely occur. Therefore, group learning plays a significant role in the development, modification, and reinforcement of mental models" [10]. The definition of shared understanding is furthermore based on a "meaning in use" point of view, which refers to coordinated action based on some resource being possessed jointly by several people. This means that it is a necessary but still insufficient prerequisite for each stakeholder to know how other disciplines think and work, and recognise where different understanding occurs (mutual understanding) in order to reach shared understanding.

However, mutual understanding does not yet mean that group members share a common viewpoint or are able to act in a coordinated manner. As our definition of shared understanding involves a "meaning in use" aspect, mutual agreement on one perspective is thus necessary to achieve shared understanding. For example, it is not enough to have a collection of requirements the different stakeholders hold, since in the course of development not only differences and conflicts among those requirements may hinder goal directed action but also different actors may prioritize and omit different requirements in their activities. The development team needs to negotiate and agree on a shared and non-conflicting mental model they want to follow.

Briggs et al. [13,11] and Kolfschoten et al. [11] distinguish between five potential sources of disagreement in collaborative requirements engineering. Three of these (differences of meaning, mental models and information) fall into the core of our concept of shared understanding, as they refer to a lack of mutual knowledge, beliefs or assumptions. They are mainly related to a certain proposal or proposal-outcome judgement [13,11]. "Differences of meaning occur when the same words or labels are used for different concepts or when different words or labels are used for the same concept" [11]. Differences of mental models occur on the level of cause and effect chains rather than on individual concepts. Both can be based on knowledge, beliefs and assumption, whereas differences of information are defined as conflicting knowledge or knowledge that not all of the stakeholders have.

When these sources of disagreement are revealed through asking clarification questions and communicating different views, mutual understanding evolves. If stakeholders agree on a common perspective on meaning, information and mental models, a shared understanding can be reached. The other two sources of disagreement are about conflicting goals or taste and might require other consensus building strategies that focus on negotiation rather than on clarification, as they exist due to differences in outcome-instrumentality judgments [13,11]. They do not result from differences in understanding, but mutually exclusive individual goals that hinder stakeholders from committing to a group goal or action.

A lot of effort has been spent on providing techniques to enhance shared understanding in the requirements engineering activities (see [14] for a discussion of the contribution of different representations to the RE activities). For example, goals [15], application scenarios [16-18] and requirements negotiation with EasyWinWin

[19,20] are proposed to support a shared understanding between stakeholders. We focus on the effectiveness and results of the combination of these three techniques to clarify the requirements in a multidisciplinary project team. There is some effort in the community to categorize and detect clarification events in written communication about requirements [21]. In our research, we distinguish between different types of clarification questions to get an idea if, and how, a mutual and shared understanding is reached, as well as which sources of disagreement are revealed.

3 Research Method

A case study to investigate the emergence of mutual and shared understanding in the system development process was performed in the research project VENUS. In this case study, a project was carried out in which a multidisciplinary team developed the mobile application Meet-U. This is depicted in the next section, followed by a description of the multidisciplinary project team. The development process including the approaches to fostering mutual and shared understanding is further shown. After the description of the case study, we describe the data collection and data analyses.

3.1 The Mobile Application Meet-U

In the case study the development of the mobile application Meet-U was attended. The idea for Meet-U had already been developed and realised in a technically oriented prototype [22]. The goal of Meet-U is to support users with regards to organising and arranging meetings with their own friends. Meet-U assists them in planning meetings or events on the way to the location or even at an actual meeting or event.

In greater detail, users can register for public events or create private meetings to which they can invite other people. Further, users can provide personal information about themselves or their interests in order to receive recommendations for events and other users with similar interests. If a user would like to attend a public event, Meet-U creates recommendations using the provided data and interests upon request. When creating private events, Meet-U recommends contacts upon request that are determined by using the settings for the event, as well as the personal interests listed by the users. Depending upon the current location of the users, they are reminded of the beginning time of the event. In addition, Meet-U provides navigation services. On-site, the event host can offer services that Meet-U recognises and integrates into the graphical user interface, such as ticket services or site plans.

3.2 The Multidisciplinary Development Team

For the development of Meet-U, socio-technical concerns and requirements [23] should be taken into account. They are related to legal conformance, usability and trust. Legal conformance refers to the inclusion of legal requirements. Usability wants to ensure that users can handle and interact with the application. Trust refers to the

intention or willingness of users to be vulnerable to important actions of the system without the ability to monitor or control the system [24].

To consider the socio-technical requirements, a multidisciplinary development team consisting of four developers and three domain experts was formed; more precisely, a legal expert, an expert for perceived user trust and user acceptance, and a usability expert were involved. The most experienced developer was responsible for the management of the project. The first author functioned as an observer in the development team and attended the project meetings. The team members had known each other for at least one year due to the cooperation in the research project.

3.3 Development Approach

The development of Meet-U took place from October 2011 until April 2012 (there was a four week Christmas break). To assess the socio-technical requirements, the whole development was carried out by the multidisciplinary team: demand analysis, requirements engineering, conceptual design, software design, implementation and evaluation. Figure 1 illustrates the phases that are briefly summarised in the following sections (see Comes et al. [25] for details and a discussion of the results regarding the development approach). Due to the fact that the development was integrated in a research project, the requirements were repeatedly reflected upon and discussed anew by the development team until September 2012.

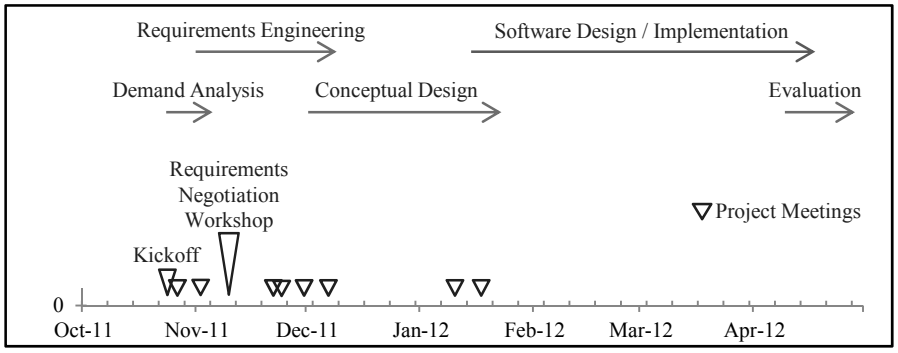

Fig. 1. Phases of the Development Project

In order to enable the collaboration of stakeholders in the first phase of development beginning with a kick-off on the 25th of October 2011, the team created goals [15] and application scenarios [16] to establish an interdisciplinary vision of the mobile application. Scenarios are a particular kind of design artefact intended to facilitate shared understanding of the target system, its interaction with users and subject domain, and its larger context [17]. Goals and scenarios are widely used in requirements engineering as a common basis for communication, and are well suited to resolve misunderstandings with stakeholders from different disciplines [26,18]. They also enforce interdisciplinary learning [18]. Therefore, the application goals

were outlined from the perspective of users, after which they were refined for the application scenarios.

Further, persona were created as archetypical representatives of user groups in order to make the scenarios as realistic and comprehensible as possible for all involved stakeholders with specific, future users. In an additional activity, a business model was developed as an extension to the scenarios in order to assess the marketability. A validation of the extended scenarios was carried out with potential users to reveal incorrect assumptions about users and the application. Later, the scenarios were used as a reference by stakeholders during the development project in order to retain focus on the goals selected from the user perspective.

In requirements engineering, the stakeholders collected, analysed and documented the requirements. A computer assisted requirements negotiation workshop following EasyWinWin [19] was used to agree upon all requirements that were collected in advance. EasyWinWin "is based on the WinWin requirements negotiation model and helps a team of stakeholders to gain a better and more thorough understanding of the problem and supports co-operative learning about other's viewpoints" [20]. The workshop took place at the 10th and 11th of November 2011.

In a first step, the stakeholders evaluated the comprehensibility of the requirements, created a glossary of terms and definitions, and adjusted the requirements. The requirements deemed important by one stakeholder were transferred to a new list if all stakeholders agreed that they had understood the requirement (in order to avoid redundancies). In accordance with EasyWinWin, the requirements were then rated by the stakeholders in terms of importance and ease of realisation. In the next step, stakeholders could express concerns regarding certain requirements in the tool. In another round, proposals for solutions for the issues were collected, before a conjoint agreement was reached by means of a group discussion. After the requirements negotiation, the requirements were structured and added to the requirements documentation.

In the concept design, different kinds of design artefacts intended to facilitate shared understanding were used. First, use cases were developed. The multidisciplinary team verified the use cases in order to ensure a correct requirement transformation. In the second step, the data and functional elements of the application were described. Thus, all information provided for the user and every operation the user could make were identified. Flowcharts were employed to graphically illustrate the operation steps and the corresponding data and functional elements. Further, the structure of the user interface was depicted in a sitemap. The fourth step consisted of deriving a first graphical design with a functionless prototype of the user interface. All stakeholders received the produced artefacts and were asked to check if the requirements had been fulfilled.

The resulting artefacts, agreed upon in an interdisciplinary manner, functioned as a working basis for the developers in the implementation phase. The application concept was implemented in an iterative process. Next, the created prototypes were assessed by experts with regards to whether the previously defined requirements were taken into account during the realisation. This examination enabled changes to be made to the application concept that were integrated into the next iteration. In

addition, the component functions developed in the process were evaluated from a user perspective.

The concluding evaluation of the usage aimed at assessing the functionality as well as the social compatibility of the system. It was experimentally tested with real users in as realistic application surroundings as possible in order to see whether the requirements had been fulfilled. See Söllner et al. [27] for more information concerning the realisation and selected evaluation results.

3.4 Data Collection

In order to analyse the communication in the development project, quantitative data collection and evaluation methods were selected. We conducted a document analysis for the collection of data. The objects of investigation were: the description of the application scenarios in six versions; the business model in three versions; the list of requirements in six versions; four versions of the use cases; the workflows and screens designs in four versions; as well as minutes of the ten project meetings. All documents as well as complementing communication were exchanged in 611 emails between members of the development team for the duration of the whole project using a project specific mailing list. These emails were the data basis for our assessment. The documents contained, apart from the actual content, distinguished changes of the pre-version, as well as comments and notes made by the involved stakeholders. The project language was German. During the collection of data, the first author functioned as an observer in the meetings of the development team.

3.5 Data Analysis

The evaluation of the documents was accomplished with the aid of a quantitative content analysis. To reduce the amount of data, the 611 emails were screened through, and relevant emails with development artefacts or textual contributions were extracted. The 183 resulting emails and documents were transformed to PDF files and stored in ATLAS.ti 6.2, providing support for manual qualitative coding. As we were interested in the emergence of mutual and shared understanding, we conducted the data analysis in three steps.

In the first step, 330 comments (one or more sentences from emails or annotations of the documents) were marked that contained questions, raised issues or indicated different understandings about a requirement. We refer to these comments as *questions* in the remainder of the text. One of the authors marked the questions in the ATLAS.ti by reading all emails twice.

In the second step, we analysed the questions of the team members. To distinguish the questions, we used the classification that was proposed by Watts et al. [28] to classify questions of understanding according to the periods in the process of conceptual change. Conceptual change occurs when participants either consolidate their current understanding, explore beyond their current knowledge to expand it or elaborate on it to challenge and test their framework of understanding [28]. Consolidation, exploration and elaboration are all indicative of changes in the current

conceptual thinking of the person asking those questions. For elaboration question that reconcile different understandings or resolve conflicts, in the third step, we used subcategories containing the key sources of conflicts proposed by Briggs et al. [13] and Kolfschoten et al. [11]. The subcategories are differences of meaning, differences of mental models, differences in information, mutually exclusive individual goals and differences of taste (Table 1).

Table 1. Categories and subcategories used for coding

Category	Subcategory	Explanation
Consolidation	-	Confirm explanations and consolidate new ideas (mutual understanding)
Exploration	-	Seek to expand knowledge and test constructs (mutual understanding)
Elaboration		Reconcile different understandings, resolve conflicts (shared understanding)
	Differences of meaning	The same words are used for different concepts or different words are used for the same concept
	Differences of mental model	Different understandings of the means for achieving desired outcomes, or of sequences of cause and effect
	Differences in information	stakeholders do not have the same information, or one stakeholder has information that other stakeholders do not have
	Mutually exclusive individual goals	Difference of interests or values
	Differences of taste	There is no rational conflict of stakes or values but rather one of taste

Two graduate students coded the questions according the categories with ATLAS.ti. They were provided with explanations and examples and received 30 minutes of training. For the coding, one student needed 5 hours and 15 minutes and the other needed 5 hours and 30 minutes. The students could, and did, ask the first author if they faced difficulties. Questions that were not assigned to the same category by both students were discussed and assigned to a category by two of the authors.

4 Results

This section reports the number of questions assigned to the different categories and subcategories. We divided the development project into three stages that are important for mutual and shared understanding in requirements engineering: the stage before the requirement negotiation where the scenario is developed and the requirements are collected, the requirements negotiation workshop which is designed

to reveal misunderstandings and reach an agreement about the system and its requirements, and the time after this agreement. In the next section, we first report the results of the assignment to the categories consolidation, exploration and elaboration. The elaboration questions are further analysed in the second subsection.

4.1 Questions for Consolidation, Exploration and Elaboration

To analyse the emergence of mutual and shared understanding, we categorised the questions and pointers in the documents according to the periods in the process of conceptual change. Questions for consolidation and exploration indicate a lack of mutual understanding; questions for elaboration indicate a lack of shared understanding.

Table 2. Questions before, during and after requirements negotiation (RN)

	Before RN	During RN	After RN	Total
Consolidation	20	24	74	118
Exploration	16	34	54	104
Elaboration	22	51	35	108
Total	58	109	163	330

Table 2 shows that there are a similar number of questions in each category and that one third of all questions were raised in the requirements negotiation workshop. Further, most conflicts could be elaborated upon before the end of the requirements negotiation, but there were more questions regarding the mutual understanding after requirements negotiation than there were in the combined before and during the requirements negotiation.

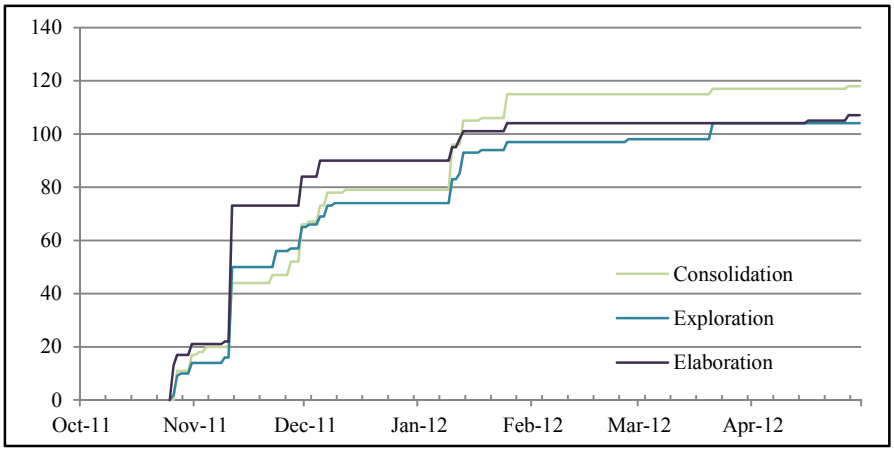

Fig. 2. Cumulative quantity of questions according to the process of conceptual change throughout the development project

Figure 2 shows the emergence of questions regarding mutual and shared understanding. Especially in late November, December and January, after requirements negotiation (including a four week Christmas break), team members raised questions for consolidation and exploration almost continuously.

This indicated that the stakeholders had the same goal but understood the requirement differently. This conflict was assigned to the category difference of mental model. The incidents of lacking shared understanding/differences in mental models concerning the requirements were especially critical, as system specification and development had already been executed at this point in time, based on the requirements, which had been agreed upon but had obviously not been fully understood.

4.2 Elaborated Conflicts

To analyse the conflicts that were revealed in the development project, we categorised the elaboration questions according to their key differences. We found that most conflicts during the whole development project dealt with different goals of stakeholders that, in most cases, were connected to their disciplinary background. For example, the legal expert wanted the user to agree on every function using personal data. In contrast, the usability expert did not want to interrupt the user while executing a task with the application. Almost the same quantity could be identified for the differences of mental models. Fewer conflicts belonged to differences of meaning, conflicting information and differences of taste (Table 3).

Table 3. Elaborated conflicts before, during and after requirements negotiation (RN)

	Before RN	During RN	After RN	Total
Differences of Meaning	5	3	4	12
Difference of Mental Model	12	8	17	37
Conflicting Information	1	5	2	8
Mutually Exclusive Individual Goals	3	33	8	44
Differences of Taste	1	2	4	7
Totals	22	51	35	108

Most conflicts regarding goals were elaborated in the requirements negotiation workshop, but differences of mental model were mostly revealed later in the project, which is critical, based on our assumption that revealing conflicts in the proposal-outcome judgement should be the basis for all further negotiation.

Figure 3 shows that differences of mental models were revealed throughout the project. Considering the differences of meaning, most conflicts were revealed before the requirements negotiation workshop; however, similar to the conflicting information and differences of taste, there were no peaks throughout the development project. Therefore, the number of conflicts remained at a low level, in contrast to the differences of mental models and individual goals.

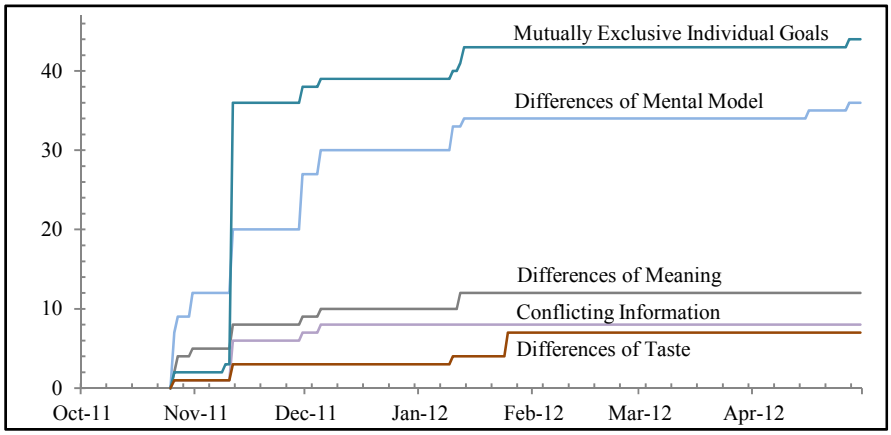

Fig. 3. Cumulative quantity of conflicts revealed throughout the development project

Summarizing, a revised shared understanding evolved late in the development phases. This led to adjustments of the artefacts and, thus, hampered the development process.

5 Discussion

The aim of our study was to analyse extent and forms of (lacking) mutual and shared understanding and how this understanding emerges in the system development process. Further, we wanted to examine which forms of conflicts occurred and in which stages of the development process they were revealed. This section discusses the results and provides suggestions for the improvement of mutual and shared understanding in development projects.

We first checked the questions according to the process of conceptual change. We could find an almost equal number of questions regarding consolidation, exploration and elaboration. As shown in the results section, the mutual and shared understanding emerged together. There were a lot of elaboration questions among the requirements negotiation workshop, but questions regarding mutual understanding emerged evenly distributed in the project. Due to the fact that a lack of shared understandings can only be detected effectively if a mutual understanding exists, there should be additional effort made in the beginning of the development project that would foster mutual understanding of the multidisciplinary team [3]. This could be done, e.g., by enforcing reflection and actively introducing techniques for construction and co-construction of meaning [29]. Bittner et al. [12] present a first attempt to develop reusable techniques for systematically building mutual and shared understanding.

To strengthen this stream of research and enlarge the set of available techniques, further research into understanding and designing mutual and shared understanding is thus necessary. In requirements engineering, natural language software requirement patterns [30-32] could help to foster a mutual understanding by using standardised, well defined and discipline independent terms and formulations. Further, the

unambiguity could be fostered with a proven template that is provided by the requirement pattern.

The investigation of the elaboration questions indicated that all five types of conflicts occurred in the development project. This goes in line with Briggs et al. [13] and Kolfschoten et al. [11]. Further, we could quantify the different categories. Most conflicts belonged to the categories' mutually exclusive individual goals und difference of mental model. While the requirements negotiation workshop was good at revealing mutually exclusive individual goals, it was insufficient for revealing differences of the mental model. Over the time of the project the differences of the mental model emerged continuously, only fostered by repeated interactions of the stakeholders. Together with the observation that there was also no concentration of consolidation and exploration questions in the requirement negotiation workshop, we assume that EasyWinWin helps to deal with conflicting goals of the stakeholders, but other approaches are necessary to foster other problems in understanding.

These issues - important to address as artefacts in the development process - are highly interrelated and build on each other. Late changes of requirements due to differences in meaning or mental models, which should have been detected and clarified early in the process, might require new negotiation efforts on goals or taste when the system has already been agreed on. We assume that in an effective requirements negotiation process, differences of understanding should be discovered as early as possible, as mutual understanding is a prerequisite for shared understanding.

Based on mutual understanding, a shared perspective can be negotiated. Shifts in this process of detecting and resolving sources of disagreement might require unnecessary iterative loops and delays. Thus, collaboration techniques should be applied to shift those attempts from coincidence to a systematic and reusable process. For this purpose, group model building techniques can be used or analysts should search for conflicting assumptions behind the conflicting models [11]. A lack of shared understanding caused by differences of the mental model might also be addressed with software requirement patterns. Apart from the proven formulation of the requirement template, they can provide background information that helps other stakeholders understand the causes and estimate the effects of the requirement.

6 Limitations

This section summarises the threats to the validity of the work.

The internal validity of the case study could be threatened by the fact that we analysed only the written communication (including the annotated development artefacts) in the project and minutes that were taken in the meetings. The requirements negotiation workshops and the meetings were conducted in the presence of the observer but without recording of the oral communication. In the requirements negotiation workshop, the stakeholders were encouraged to write down their questions and issues through the use of the computer-based EasyWinWin. Thus, we could analyse them in detail. Although we did not prevent oral communication in the

workshop or in other meetings, the focus on the written communication is a limitation of this study.

Coding the data analysis, the students reached agreement on most questions but were also faced with difficulties. Especially questions that were asked very politely to show (in subsequent discussion between the stakeholders) that there might be a conflict. These were partly assigned to consolidation or exploration. Also, they had some difficulties with questions that consolidated new ideas. If they read a question alone they had difficulty deciding if it was just a new idea or a conflict. To clarify this, the students could consult the first author that observed the development project and had attended the project meetings. All questions with such uncertainties were discussed by two of the authors before they were assigned to categories. Therefore, background knowledge of the development project was partly necessary to assign some of the questions.

Regarding external validity, the major concern is the generalizability of the results since we conducted only one case study. The case study with seven people is embedded in a research project that has distinct features such as the repeated discussion and reflection about requirements, which might have an impact on the emergence of the shared understanding. Due to the diversity of the development and requirement engineering approaches, we cannot claim that the results are representative for all development projects. Further, the team and stakeholders involved with their different backgrounds could have had an effect on the emergence of mutual and shared understanding. This study is a first step to analyse the emergence of mutual and shared understanding. To strengthen the results, other development teams with stakeholders from various disciplines should be analysed.

7 Conclusion

In this paper we analysed the emergence of mutual and shared understanding in the written communication of a multidisciplinary team that developed a mobile application. The team used application scenarios and an EasyWinWin requirement negotiation workshop to reveal and overcome a lack of understanding. We showed that the workshop helped to identify most conflicting goals of the stakeholders, but differences in the mental model were mostly identified in other stages of the process. Further, consolidation and elaboration questions belonging to mutual understanding were equally distributed in the process. Hence, we could not observe an effect by the requirement negotiation workshop. Even when artefacts were already agreed upon, the development team explored lack of mutual understanding to underlying concepts or relationships. If a shared understanding in the development team is important, there should be additional approaches used in requirement engineering activities.

This paper has several implications for research. We used a classification for mutual and shared understanding based on the process of conceptual change. This approach can differentiate the success of clarification techniques based on different types of understanding and can be used to get a deeper understanding of project communication. The results show that in our case study the requirements negotiation

workshop worked well for most things but not for the crucial issue of different mental models. This indicates, on the one hand, the suitability of this requirements negotiation technique, but, on the other hand, calls for other techniques to build shared mental models. Future work should examine whether these observations can also be done in other settings.

In practice requirements, analysts should be aware that a lack of understanding can have different sources and that RE techniques are more or less suited to address the different types of mutual and shared understanding. If an agreement by stakeholders shall be reached, requirement analysts should spend effort to achieve a mutual understanding of the requirements and a shared mental model of the planned system before other kinds of conflicts are elaborated upon.

To foster mutual and shared understanding in interdisciplinary projects, we call for future research to analyse extent and forms of (lacking) mutual understanding in other development projects consisting of stakeholders from various backgrounds and using various development approaches. Further, we call for research that explores ways to systematically build upon this understanding.

References

1. Tan M (1994) Establishing mutual understanding in systems design: An empirical study. Journal of Management Information Systems 10 (4):159-182
2. Aranda G, Vizcaíno A, Piattini M (2010) A framework to improve communication during the requirements elicitation process in GSD projects. Requirements Engineering 15 (4):397-417. doi:10.1007/s00766-010-0105-9
3. Corvera Charaf M, Rosenkranz C, Holten R (2012) The emergence of shared understanding: applying functional pragmatics to study the requirements development process. Information Systems Journal 23 (2):115-135. doi:10.1111/j.1365-2575.2012.00408.x
4. Berkovich M, Leimeister J, Hoffmann A, Krcmar H (2012) A requirements data model for product service systems. Requirements Engineering (online first):1-26. doi:10.1007/s00766-012-0164-1
5. Baxter G, Sommerville I (2011) Socio-technical systems: From design methods to systems engineering. Interacting with Computers 23 (1):4-17. doi:10.1016/j.intcom.2010.07.003
6. Kleinsmann M, Buijs J, Valkenburg R (2010) Understanding the complexity of knowledge integration in collaborative new product development teams: A case study. Journal of Engineering and Technology Management 27 (1-2):20–32
7. Vreede G-Jd, Briggs RO, Massey AP (2009) Collaboration Engineering: Foundations and Opportunities. Journal of the Association of Information Systems 10 (3):121-137
8. Valkenburg R, Dorst K (1998) The reflective practice of design teams. Design Studies 19 (3):249-271. doi:10.1016/s0142-694x(98)00011-8
9. Darch P, Carusi A, Jirotka M (2009) Shared understanding of end-users' requirements in e-Science projects. In: 5th IEEE International Conference on E-Science, 2009. pp 125–128
10. Mohammed S, Dumville BC (2001) Team mental models in a team knowledge framework: Expanding theory and measurement across disciplinary boundaries. Journal of Organizational Behavior 22 (2):89-106
11. Kolfschoten G, Briggs RO, de Vreede GJ (2009) A Diagnostic to Identify and Resolve Different Sources of Disagreement in Collaborative Requirements Engineering. In: International Meeting on Group Decision and Negotiation (GDN), Toronto, Canada, 2009.

12. Bittner EAC, Leimeister JM (2013) Why Shared Understanding Matters - Engineering a Collaboration Process for Shared Understanding to Improve Collaboration Effectiveness in Heterogeneous Teams. In: 46th Hawaii International Conference on System Sciences (HICSS), Maui, Hawaii, 2013.
13. Briggs RO, Kolfschoten GL, Vreede GJ (2005) Toward a theoretical model of consensus building. In: Americas Conference on Information Systems (AMCIS), Omaha, Nebraska, USA, 2005. p 12
14. Sutcliffe A (2010) Collaborative Requirements Engineering: Bridging the Gulfs Between Worlds. In: Nurcan S, Salinesi C, Souveyet C, Ralyté J (eds) Intentional Perspectives on Information Systems Engineering, vol 1. Springer-Verlag, Berlin Heidelberg, p 355
15. Dardenne A, Lamsweerde Av, Fickas S (1993) Goal-directed requirements acquisition. Sci Comput Program 20 (1-2):3-50. doi:http://dx.doi.org/10.1016/0167-6423(93)90021-G
16. Haumer P, Pohl K, Weidenhaupt K (2002) Requirements elicitation and validation with real world scenes. Transactions on Software Engineering 24 (12):1036-1054
17. Jarke M, Bui XT, Carroll JM (1998) Scenario management: An interdisciplinary approach. Requirements Engineering 3 (3):155-173
18. Weidenhaupt K, Pohl K, Jarke M, Haumer P (1998) Scenarios in system development: current practice. IEEE Software 15 (2):34-35. doi:10.1109/52.663783
19. Gruenbacher P (2000) Collaborative requirements negotiation with EasyWinWin. In: Database and Expert Systems Applications, London, 2000. pp 954-958
20. Briggs RO, Grünbacher P (2002) EasyWinWin: Managing Complexity in Requirements Negotiation with GSS. In: 35th Hawaii International Conference on System Sciences, Hawaii 2002.
21. Knauss E, Damian D, Poo-Caamano G, Cleland-Huang J (2012) Detecting and classifying patterns of requirements clarifications. In: 20th IEEE International Requirements Engineering Conference (RE), 2012. pp 251-260. doi:10.1109/re.2012.6345811
22. Comes D, Evers C, Geihs K, Saur D, Witsch A, Zapf M (2011) Adaptive Applications are Smart Applications. In: International Workshop on Smart Mobile Applications, San Francisco, 2011.
23. Geihs K, Leimeister JM, Roßnagel A, Schmidt L (2012) On Socio-technical Enablers for Ubiquitous Computing Applications. In: The 12th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT), Izmir, Turkey, 2012.
24. Lee JD, See KA (2004) Trust in Automation: Designing for Appropriate Reliance. Human Factors 46 (1):50-80
25. Comes D, Evers C, Geihs K, Hoffmann A, Kniewel R, Leimeister J, Niemczyk S, Roßnagel A, Schmidt L, Schulz T, Söllner M, Witsch A (2012) Designing Socio-technical Applications for Ubiquitous Computing - Results from a Multidisciplinary Case Study. In: Göschka K, Haridi S (eds) Distributed Applications and Interoperable Systems (DAIS 2012), Stockholm, 2012. Lecture Notes in Computer Science. Springer, Berlin / Heidelberg, pp 194-201. doi:10.1007/978-3-642-30823-9_17
26. Pohl K (2008) Requirements Engineering. dpunkt-Verl., Heidelberg
27. Söllner M, Hoffmann A, Hoffmann H, Wacker A, Leimeister JM (2012) Understanding the Formation of Trust in IT Artifacts. In: International Conference on Information Systems (ICIS), Orlando, Florida, USA, 2012.
28. Watts M, Gould G, Alsop S (1997) Questions of Understanding: Categorising Pupils' Questions in Science. School Science Review 79 (286):57-63
29. Van den Bossche P, Gijselaers W, Segers M, Woltjer G, Kirschner P (2011) Team learning: building shared mental models. Instructional Science 39 (3):283-301. doi:10.1007/s11251-010-9128-3

30. Hoffmann A, Söllner M, Hoffmann H (2012) Twenty software requirement patterns to specify recommender systems that users will trust. In: 20th European Conference on Information Systems (ECIS), Barcelona, Spain, 2012. Paper 185
31. Withall S (2008) Software Requirement Patterns. Microsoft Press, Redmont, Washington
32. Renault S, Mendez-Bonilla O, Franch X, Quer C (2009) A Pattern-based Method for building Requirements Documents in Call-for-tender Processes. International Journal of Computer Science and Applications 6 (5):175 - 202