An empirical cognitive model of the development of shared understanding of requirements.

Jim Buchan

SERL, School of Computing & Mathematical Sciences, Auckland University of Technology, Private Bag 92006, Auckland 1142, New Zealand
jim.buchan@aut.ac.nz

Abstract. It is well documented that customers and software development teams need to share and refine understanding of the requirements throughout the software development lifecycle. The development of this shared understanding is complex and error-prone however. Techniques and tools to support the development of a shared understanding of requirements (SUR) should be based on a clear conceptualization of the phenomenon, with a basis on relevant theory and analysis of observed practice. This study contributes to this with a detailed conceptualization of SUR development as sequence of group-level state transitions based on specializing the Team Mental Model construct. Furthermore it proposes a novel group-level cognitive model as the main result of an analysis of data collected from the observation of an Agile software development team over a period of several months. The initial high-level application of the model shows it has promise for providing new insights into supporting SUR development.

Keywords: Requirements understanding. Distributed cognition, Team Mental Model, shared cognition

1 Introduction

There is a clear need for customers and software development teams to share and refine understanding of the client's requirements. Although activities related to this are most intense in the early phase of requirements discovery and validation, they continue throughout the software development lifecycle. Inadequate shared understanding of these requirements, or breakdowns in sharing this understanding from miscommunications and misunderstandings, can have a very high impact on requirements quality, project costs, development productivity, and application quality [1].

In software development, the development of shared understanding of requirements is most closely associated with Requirements Engineering (RE) activities. In [2], Sutcliffe describes RE as about "doing the right thing", as opposed to "doing things right" (the domain of software engineering). He points out that there is little value to a client in expertly implementing a software solution that does not address the right application domain problem. The challenge of RE is how to effectively and efficiently develop a consistent view of what the "right thing" is. In [3], Bubenko emphasizes the importance of the interactions between the system production team with organizational actors (clients) to understand their “visions, intentions, and activities regarding their need for computer support.” Development of high quality shared understanding of these needs and requirements will (iteratively) lead to a high quality solution design. It will provide solid foundation for reasoning and negotiating with clients about the characteristics of the desired system to be implemented.

In practice this is an inherently complex and challenging process, relying on a complex network of interactions and information flows. It involves people with divergent backgrounds and world-views, as well as the manipulation of a multitude of artifacts [4]. A consequence of this is that development of this shared understanding of the stakeholder requirements and the requirements of the goal software solution is often very time consuming, difficult to monitor, and prone to misunderstandings and miscommunications [4]. Some of the barriers and enablers related to the development of shared understanding of stakeholder requirements have been reported in our earlier work, confirming its complexity and challenge [5].

Given that (1) the development of a shared understanding of requirements (SUR) is a major aim of RE, (2) it is an enduring challenge and error-prone, due to its inherent complexity, and (3) it has a high impact on RE quality (and subsequent project success), there is a clear case for continuing empirical research in this area in order to understand the problems to address. Taking the lead from [6], in which it is observed that not enough research effort has been put into “advancing a theoretical or empirical understanding” of RE activities in practice and why they are so challenging, this paper advances a theory of SUR development in the context of RE based on an empirical study of practice. The aim is to deepen understanding of the practitioner’s problem so that the challenges of SUR development can be explained. This provides a defendable basis for deciding on how to address them. In addition, such an empirically informed theory provides a foundation for evaluating existing and new techniques and tools.

This paper takes a cognitive view of SUR and develops an empirical model of the cognitive activities involved in the development of SUR. Reframing the challenges of SUR development from the perspective of this cognitive framework provides a mechanism for applying principles from cognition theory to SUR development. Analyzing the empirical cognitive model using theories of cognition provides a cognitive explanation of the challenges (and enablers) of SUR development. Application of cognitive principles leads to new (cognitively-based) strategies for addressing these challenges. This paper reports the development of the empirical cognitive model of SUR evolution that is based on the analysis of field data gathered from an extended, non-participatory observational case study of a team developing software in a commercial setting. The application of principles from cognition theory to this new cognitive model is introduced in this paper, but the detailed analysis and implications are to be reported in a future publication.

The next section introduces the notion of SUR as a (dynamic) state of shared cognition with group-level properties appropriated from the Team Mental Model construct. This idea is extended in section 3 to develop a high-level cognitive view of SUR development that identifies two phases of cognition: monitoring for gaps in

SUR, and addressing any gaps uncovered. This perspective shapes the subsequent data collection and the content and interaction analyses of the field data, which are described in section 4. The detailed model of the cognitive activities involved in SUR development that emerges from the data analyses is described and discussed in section 5. The conclusion and future work are presented in section 6.

2 SUR as requirements-focused Team Mental Models

In order to know what data to collect and how to analyze this data to understand the challenges of SUR development, the SUR construct needs to be clearly defined. The idea of a “shared” understanding implies that the understanding is inherently a group-level property, since it cannot be a property of an individual alone, but it is an ambiguous and contentious term. What cognitive structure is it that is shared between the individuals in a state of shared understanding? What is it that changes when shared understanding evolves? What does “shared” mean in this cognitive context? Identical? Consistent? Overlapping? Compatible? Following the lead of [7], who argue for the importance of being explicit about the meaning of “shared understanding” being adopted in related research, this section provides a working definition of the SUR construct before applying it to develop a high level model of SUR development in the next section.

A well-established construct from studies of team work is the Team Mental Model (TMM) [8], a form of shared cognition. In literature the TMM construct emerged to help understand how teams work in complex, dynamic and uncertain contexts. Empirical studies of team work have shown that high levels of convergence of team members’ mental models are causally linked to high levels of team performance. The same goal and contextual characteristics apply to the study of collaborative software development. It is therefore reasonable to view SUR as a specialized form of TMM, with a requirements focus. Taking this view, a state of SUR (at some point in time) is attributed with the same properties as a TMM and so, adapting the description in [8], can be conceptualized in the following way.

(1) SUR is viewed as structured mental representations of knowledge and understanding about relevant aspects of requirements, that are similar in each team member [9].
(2) SUR is considered an enabler of a team’s effectiveness by providing a mechanism for team members to be on the same page in the sense of describing, predicting and explaining requirements in a similar way [9].
(3) SUR is conceptualized as emerging states of the team with group-level properties shaped by the cognitive contribution of team members, but more than an aggregation of their individual requirements understanding [10].
(4) The content of SUR is shared knowledge structures that include declarative (what), procedural (how) and strategic (why) knowledge about requirements [9].
(5) The property of “sharedness” in SUR is conceptualized as cognitive similarity and is the degree to which team members’ understanding of requirements are

similar in the sense of having some common or overlapping (but not identical) knowledge structures that are consistent [11].

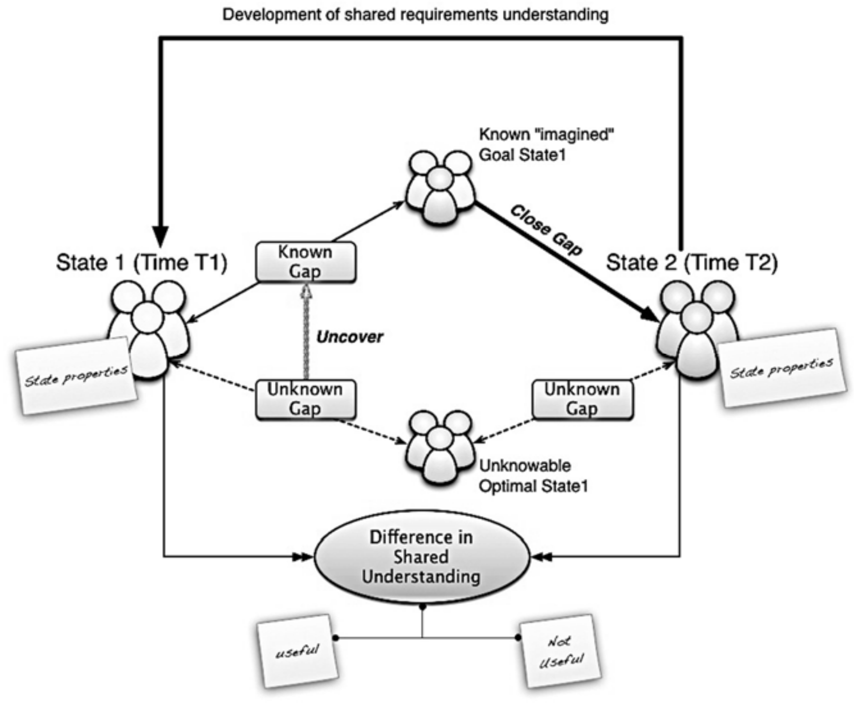
(6) SUR has the property of “accuracy”, which refers to how closely the SUR aligns with the “true state of the world” [12].

The notion of a snapshot of SUR at some point in time as a state of shared cognition (similar mental models) with the specific group-level properties of content, “sharedness” and accuracy provides a useful conceptualization of the SUR construct. The question now is how is new SUR created? What is the mechanism that results in the team developing successively higher levels of useful shared understanding of the stakeholders’ requirements? With the view of SUR as a state (set of properties) of the group at some point in time, it is natural to consider the emergence and development of shared understanding of requirements as a dynamic move through a sequence of states in “shared requirements understanding” space. Taking this view, a group’s shared understanding of a requirement changes from one state to another as the group work jointly on improving and sharing this understanding. This idea is discussed in the next section to provide a high-level framework of SUR development that shapes the subsequent fieldwork and data analysis to develop a more detailed empirical model.

3 SUR development as dynamic state transitions

Figure 1 presents model of SUR development based on the notion of SUR evolving in time from one state of SUR to another. Framing SUR development in this way highlights the notion of a gap between the two states of SUR and suggests that a mechanism is needed to address this gap. This identifies two main high-level activities in SUR development: (1) the collaborative uncovering of a gap (i.e. a level of insufficiency) in SUR, and (2) collaboratively addressing this gap to achieve a new state of SUR. The constant uncertainty in sufficiency of shared understanding discussed in literature is depicted by a constant gap in the current state of shared understanding and some idealized (unknowable) optimal state of shared understanding, where all necessary, sufficient understanding is shared and accurate for the tasks in hand at that point in time. Uncovering a gap in SUR is conceptualized as collaboratively designing a new goal state of SUR that highlights the shortcomings of the current state of SUR. Addressing this gap in SUR involves undertaking appropriate activities to achieve this new goal state. The model shows that the team may end up in a state of SUR (at time T2) with different properties to the envisioned goal state. The degree of change of SUR may vary depending on the time frame (T1 to T2).

In this specialized TMM model, the properties of a state of SUR that may change from one state to the next include: (1) content, such as the relevant application domain knowledge structures and level of detail that is similar across team members, (2) the level of consistency of the content between team members (“sharedness”), (3) the accuracy of the content (it’s consistency with structure in the world). In this view, a gap in SUR could be relevant knowledge about a requirement that is: missing, lacks sufficient detail, is not adequately shared between team members, is inconsistent between team member, or is an error (inconsistent with the world).



Development of shared requirements understanding
Known "imagined" Goal State1
Close Gap
State 1 (Time T1)
Known Gap
State 2 (Time T2)
Uncover
State properties
Unknown Gap
Unknown Gap
State properties
Unknowable Optimal State1
Difference in Shared Understanding
useful
Not useful

Figure 1. SUR development as group-level state transitions

The model also captures the idea that the actual change in understanding transitioning from one state to the next may be useful to the team or not, in the sense that the new shared understanding of a requirement in state2 may end up being unimportant for the tasks at hand.

The model incorporates the notion of the constant pressure of the team to uncover "what they don't know they don't know" about the requirements and transform it into a known gap in their shared understanding. Then to address this gap through further collaborative effort to converge on a new negotiated state of shared understanding about a requirement.

In this model, ideal development of shared understanding would be a set of transitions where the imagined goal state, the idealized optimal state and the actual new state 2 (at time T2) are congruent, and the effort to develop shared understanding would be sufficient and necessary to transition to (ideal) state2.

Framing the development of SUR in the perspective of this model provides some new insights into potential high-level strategies for supporting the development of SUR and improving RE. These are listed below together with (in italics) related activities observed in the case study team.

(1) Creating sufficient opportunities to check for gaps in current shared understanding of a requirement. *Using an Agile RE process where these opportunities happen frequently as part of the way work is done without unacceptable extra cost.*
(2) Using appropriate strategies for transforming unknown gaps (don't know what they don't know about a requirement) into known gaps (know what they don't know about a requirement). *Uncovering assumptions, insufficient detail or missing knowledge about a requirement. Verifying representations of shared knowledge about a requirement against the real-world context of that requirement. Describing, explaining and predicting aspects of requirements in a collaborative team context*.
(3) Applying techniques that promote the alignment of the imagined goal state of shared understanding with the idealised actual optimal states. *Taking time to analyse options in framing a requirement, switching views of a requirement between problem and solution space, maintaining "memory" of previous shared understanding negotiations and outcomes*.
(4) Minimising wasteful effort that results in new shared understanding about a requirement that is not used in the design and implementation of that requirement. *Identifying and focusing on understanding the significant knowledge about a requirement and its context for the task at hand (and not wasting time on refining and sharing unimportant knowledge about a requirement).*
(5) Utilising techniques that effectively and efficiently recover from gaps in shared understanding (i.e. evolve to the next known optimal state). *Collaborative knowledge creation, knowledge seeking and knowledge filtering related to a requirement. Discussing multiple perspectives on a requirement to converge on a negotiated common understanding of a requirement. Observing or measuring real-world phenomena related to a requirement.*

In summary, the model in Figure 1 provides a high-level view of the mechanism of the development of SUR as a collaborative effort firstly to uncover gaps in SUR and then to recover from these gaps. In this TMM perspective states of SUR are distinguished by having different group-level properties of content, "sharedness" and accuracy, and gaps in SUR can be thought of as inadequacies in these for the tasks at hand. The TMM view of SUR suggests that collaborative activities such describing, explaining, and predicting aspects of requirements may uncover gaps in SUR and, by addressing these, create a new state of SUR. Finally the model suggests some high-level strategies for supporting the development of SUR, supported by observation of practitioners at work.

These initial insights are used to analyze the data gathered from the in-depth case study by focusing on identifying patterns of interaction during collaborative RE activities that involve uncovering a gap in SUR and recovering from this gap. The context of the case study, the research approach and the data collection and analyses, are discussed in the next section.

4 The research approach

The high-level aim of this study is to gain an understanding of how shared requirements understanding emerges, develops and is maintained in the context of a team undertaking collaborative software development, with a view to suggesting strategies for supporting its development. The research approach is to develop a detailed local account of the phenomenon by observing a team as they collaboratively develop software. This is achieved by analyzing step-by-step how such shared understanding develops from an initial lack of shared understanding to successively clearer shared understanding, as well as how misunderstandings are collaboratively uncovered and recovered from. The level of analysis is the team, as part of a distributed cognitive system. This is grounded in the TMM conceptualization as well as arguments put forward in [13] for a new science of Group Cognition, in the notion of a functional system as the computational engine of Distributed Cognition described in [14], and in the emergence of collective intelligence presented in [15]. The development of the cognitive model presented is based on thematic content [16] and interaction analyses [17] of the data. This approach is similar to that taken by Stahl ([18]) in his analyses of online data of teams solving mathematics problems where he emphasizes the centrality of team members' interactions to the development of group cognition.

4.1 The case study organisation

A non-participative in-depth case study methodology was adopted [19] with data collected by the author over a 5-month period in a single case organization. The single case study research methodology is appropriate to the research aim of developing an in-depth understanding of the collaborative work of practitioners in their place of work, and all the complexity this involves [20]. The organization is in the Financial/Insurance sector and of the five organizations invited to participate, they were selected based on their large in-house software development program, their use of Agile methods, and their willingness to participate. The study involved observation of an in-house development team collaboratively developing software for both in-house and external clients using a customized Agile process. The project involved the improvement of existing systems and the implementation in sites geographically spread throughout New Zealand. Ethics approval for the research was granted and participants voluntarily signed consents to participate and be recorded, prior to the start of the study. Participants had the option of opting out of the study at any time during the study, although this situation did not arise.

The core team consisted of three business analysts (BA), two legacy system developers, a Java developer, two testers, a project manager (PM), and two Product Owners (PO). The POs had their own managerial jobs in addition to being POs for this project and did not sit in the team's work area. They were generally available for relevant meetings, however. The PM acted as a team mentor and liaised with higher management and other development teams doing related work, as well as doing some long-term resource and roadmap planning. Two subject matter experts (SMEs) from customer services and an architect were closely associated with the project, but not full-time, and did not sit in the team's work area. The SMEs were a source of requirements (from customers) and elaborated stories, described process constraints and argued for prioritization of certain requirements. The architect was often consulted (or stepped in) to provide advice regarding system and design constraints, business processes and explain historical decisions related to these. The architect had the authority to veto a user story or and its proposed implementation design. Other SMEs and managers were involved regularly (for example the National Sales manager, the Communications manager,) and others as-needed. There was no Scrum Master role identified, although one of the BAs had the role of Sprint Coordinator and often acted as Scrum Master. The experience level of the team members in using Agile practices varied from novice to expert, and some training in the implementation of Agile ideas was provided during the period of observation. In addition while I was there, the team had three afternoons set aside to discuss how to make Agile work for them in this project.

The development process followed a customized Agile process using a Scrum framework with 4-week sprints, daily stand-up Scrum meetings, sprint planning meetings, sprint reviews and retrospectives. Before each Sprint there were sprint planning meetings to elaborate, prioritize and estimate requirements. Planning poker was used to negotiate effort estimations of user stories. Scrum meetings were typically between 5 and 10 minutes and anyone could attend them, although only the core team and project stakeholders could speak. They kept track of who is talking by passing a ball around and generally postponed any lengthy discussions of problems identified until after the Scrum meeting.

User stories were the main representation of requirements and these were generally written on physical cards in the common "As a…I want…so that…" structure. The user stories were duplicated on an Excel spreadsheet, although included more detail than on the cards. Some team members also duplicated the user story cards in Jira. A story board was at the front of the work area, visible to the core development team members at all times. Scrum meetings took place in front of this board. The board had columns representing the development workflow during a sprint. Other information on the story board included: the sprint goal, a delivery roadmap, a vision statement, the definition of "done", and a sprint burn-down chart. The story board was personalized with a theme selected by the team and on the board team members were represented by pictorial avatars related to that theme. Acceptance tests were designed and run by the two testers who worked closely with the developers and BAs. Maintenance and support of implemented features were handed over to a separate department.

4.2 Data collection

During the field work the author was stationed at a desk in the team's main work area and attended many of the team meetings, but did not participate in discussions. Fieldwork was interleaved with some data analysis and other academic duties outside the organization. Typically the fieldwork was between 10 and 30 hours per week depending on the researcher's other duties. The software development lifecycle observed in the study included pre-project work with some team members still involved in an earlier project, pre-sprint activities, and three full sprints of four weeks each.

The data were collected in the form of extensive field notes (meeting details, observations and ideas), as well as photographs and electronic recordings (audio and some video) of many of the formal and informal meetings and the work area. The field notes, media files, and documents were cross-indexed so that all the data related to any specific meeting, meeting type, date, team role, or location could easily be identified. The audio and video recording were kept unobtrusive by discreetly using an iPad as the recording device. Important artifacts were collected or photographed. Around 100 hours of audio and video were collected and a subset of these was transcribed for further analysis. Meetings observed and recorded included meetings for story prioritization, story elaboration, story estimation, management updates, inter-team updates, sprint planning, roadmap planning, organizational strategic planning, retrospectives, process understanding, team problem solving, as well as daily standup meetings, retrospectives and informal ad hoc meetings. Occasionally participants were interviewed briefly after a meeting to clarify observed behavior, provide background information or clarify the meaning of unfamiliar language. The results of the interviews were generally noted in the field notes and occasionally recorded, typically for longer interviews.

4.3 Data selection and analysis

During the fieldwork episodes observed that were significant with respect to the development of shared requirements understanding were noted in the field notes. This typically included episodes in meetings where collaborative sense-making of requirements was significant: the collaborative understanding of a requirement was the focus of sustained effort; a gap in the shared understanding of a requirement was uncovered; the shared understanding of a requirement changed significantly; the shared understanding of a requirement was socialized to a wider group; the interactions with each other and artifacts were particularly rich. The media data (audio and in some cases video) associated with the meetings containing these significant episodes were then selected for further analysis based on a judgment of their significance. Other meetings and episodes were also selected for further analysis based on the aims of: good coverage of meeting types, temporal coverage of the full lifecycle of specific user stories, coverage of phases of sprints (e.g. start and end) and coverage of role involvement.

A selection of recorded audio and video episodes (16 hours of audio and 1 hour of videos) was selected for transcription and these transcriptions were imported in to an analytical tool, NVivo. Their inclusion for transcription and subsequent analysis was based on the high-level framework of analysis in Figure 1, with the aim of including examples of uncovering different types of gaps in shared understanding of requirements and addressing these gaps, in a variety of collaborative contexts. For a specific identified episode the entire meeting containing the episode was transcribed to ensure sufficient context for interpreting interactions and content meaning. For example, a user story related to providing a feature to synchronize the billing cycles of a customer for different products purchased at staggered times was particularly challenging, and all meetings in which the shared understanding of this user story gained the team's attention were transcribed. Studying the changes to the shared understanding of this user story and the interactions involved in these changes provided a rich dataset on a micro- and macro-level involving many team roles, artifacts, meeting types and types of collaborative cognition. The transcriptions of meetings involved transcribing dialog, identifying speakers consistently (from 2 to 12 speakers in any one meeting) and time-stamping significant episodes.

In order to understand the "content" property of a state of SUR and changes to this SUR content, a content analysis of the relevant data was performed, based on the inductive approach described in [16]. The data used includes the transcription of the episode dialog, as well as any related artifacts (e.g. story cards, spreadsheets, photographs of material developed on a white board, photographs of the story board, or videos of the episode). The aim of the content analysis was to develop a representation of the team's shared knowledge structure about a requirement (the SUR content) at that point in time. A concept map [21] was selected as a suitable representation since it depicts significant concepts and their relationships in a network structure. Following the method described in [16] the transcripts of the episodes were coded and categorized (using NVivo) to identify significant concepts and their relationships related to a requirement. This content analysis has a quite a restricted aim and inclusion of the "latent content" (e.g. silence, laughing, body language, tone) was considered unnecessary. The concept map was constructed with the aid of a software tool CMapTools [22]. The concept map developed was crosschecked against other shared artifacts representing shared understanding of that requirement at that time, and any appropriate additions or modifications to the knowledge structure represented in the concept map were made. The same exercise was repeated using data from an episode at a later time and the differences in the concept maps analyzed to identify the changes in content of the two states of SUR. Lack of space precludes presenting examples of the concept maps and their analysis. The technique shows good promise as a method of depicting a snapshot of the content of a state of SUR. The concept map also proved useful as a (dynamic) representation of the application domain knowledge relevant to a specific requirement. It is also interesting to note that the knowledge structure representing a state of SUR includes technical and process knowledge as well as application domain knowledge.

In order to develop the collaborative cognitive model of the development of SUR principles from interaction analysis [17] and content analysis [16] were used. Analysis of the data followed paths of both inductive and deductive reasoning as described in [16]. The framework presented in Figure 1 was used as a starting point to provide two general categories of cognitive activity, namely uncovering a gap (in SUR) and addressing that gap. This was used deductively to gather and code content according to these high-level categories. An inductive approach was taken with the interaction analysis. The interaction analysis is concerned with both the enactment and the content of the interactive dialog and how development of SUR is achieved through this participant interaction. The interactions, as sequences of actions and speech, were coded, grouped, categorized and abstracted to develop the cognitive model depicted in Figure 2. Twenty-nine types of cognitive interaction were initially identified and coded using NVivo, grouped into the two general categories of uncovering and addressing a gap in SUR, as previously discussed. (These include, for example, proposing, questioning, persuading, reinforcing, explaining, describing, comparing, abstracting, testing and deciding). These interaction code were grouped and categorized to provide a smaller subset of key categories that were then abstracted to the key cognitive tasks and decisions presented in Figure 2 and described in section 5. Another researcher crosschecked parts of this coding and abstraction process from the data. The empirical cognitive model of the development of SUR is presented in the next section, and a brief overview is given.

5 An empirical cognitive model of SUR development

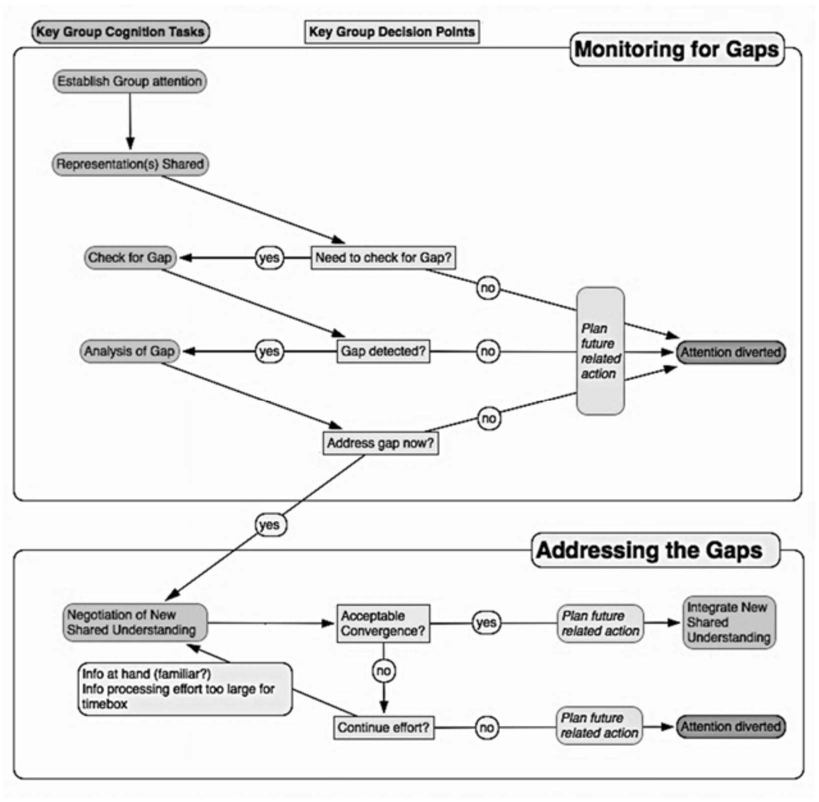

Key Group Cognition Tasks
Key Group Decision Points
Monitoring for Gaps
Establish Group attention
Representation(s) Shared
Check for Gap
yes
Need to check for Gap?
no
Analysis of Gap
yes
Gap detected?
no
Plan future related action
Attention diverted
no
Address gap now?
yes
Addressing the Gaps
Negotiation of New Shared Understanding
Acceptable Convergence?
yes
Plan future related action
Integrate New Shared Understanding
Info at hand (familiar?)
Info processing effort too large for timebox
no
Continue effort?
no
Plan future related action
Attention diverted

Figure 2. An empirical cognitive model of SUR development

The model depicted in Figure 2 presents a high-level conceptual model of the group-level cognitive tasks involved in refining SUR. It is based on patterns of interaction identified from analysis of the collected data across many collaborative episodes and RE activities. It follows the form of the state-change model in Figure 1 in terms of describing the cognitive tasks involved in monitoring for gaps in SUR and addressing any uncovered gaps.

5.1 Collaborative cognitive tasks

The significant high-level cognitive tasks identified are: (1) establishment of the team's attention on some aspect of the shared understanding of a requirement; (2) the presentation (oral or visual) of some representation of the requirement within the horizon of observation of the group; (3) the group checking for some shortcomings in the current shared understanding of a requirement (a gap) (if the need for a check is agreed on); (4) deeper analysis of the nature of the gap and how to address it (if it is agreed there is a gap) (5) negotiation of new shared understanding of the requirement (if the group decides this is possible, worth the effort, and time is available); (6) integration of the new shared understanding of the requirement (if the negotiation of understanding converges to a sufficient level).

Cognitive activities (1) to (4) are associated with monitoring and identifying inadequacies in understanding. This is predicated on finding inconsistencies between two or more shared representations of understanding. These inconsistencies may be differences in different team members' mental representations, identified when they are articulated publicly (i.e. a consistent understanding is not shared). Or it may be inconsistencies between some representations of knowledge, or observation of the world that is shared publicly. Also, there may be a gap in the sense that an information need is identified. This can be viewed as inconsistency between the current state of shared understanding (where the information gap exists) and an imagined state of shared understanding in which the new information is integrated and shared.

Cognitive activities (5) and (6) are associated more closely with recovering from a gap in understanding, such as a misunderstanding. It should be noted that this distinction is fuzzy in the sense that negotiation of new understanding may result in gaps being uncovered and vice versa. Cognitively the distinction is clearer, however. The negotiation of new shared understanding may be a simple correction of a mistake to a lengthy collaborative exploration of alternative meanings and sense-making of the application domain problem.

While there are overlaps in the activities identified in the model and the sequencing was often iterative, there are common dependencies between activities. For example establishment of the team's attention on current shared understanding occurs before the detection of any shortcomings that need to be addressed, which itself precedes negotiation of new understanding and convergence on a new interpretation of a requirement. In addition, integration of new shared understanding with the existing shared understanding in the wider project context is generally dependent on sharing and agreement on some new or deeper understanding of a requirement. If the team decides (rightly or wrongly) that a requirement is not worth attention at that moment, then the other cognitive activities shown in Figure 2 will not occur. Similarly a shortcoming in shared understanding may NOT be detected (rightly or not), halting further consideration of that requirement. If, during exploration of alternative interpretations, an information need is detected that is not immediately available, then further exploration, negotiation and integration may be postponed until this new information is obtained and shared.

Each aspect of the depicted model may have varying levels of effort and formalism during different episodes of shared understanding refinement. For example, it may be a very quick and "intuitive" decision that a specific requirement needs attention (or not) or has a deficiency (or not). Or it may be the result of considerable cognitive effort and extended interaction. Exploration of alternative interpretations and sense-making may be rapid, based on the accepted intuition of experts in the team at the time, or it may involve extensive modelling and analysis, possibly with information sought from outside the immediate group at a later time. It may be that coordinated attention is established quickly because it expected by the group in a particular meeting, or that it takes considerable cognitive effort to coordinate attention because the participants have separate agendas.

5.2 Group decision points

There are a number of decision points throughout this model where continued effort on sharing understanding of a particular requirement may be diverted, perhaps with an agreed plan to come back to it in the future.

The model captures the observed situations where:

(1) The team does not detect a gap in shared understanding during the collaborative episode, even though there is a gap. This may be because the specific aspect of the requirement that has a gap is not given attention, or that the deficiency in shared understanding is not uncovered even after some effort checking for a gap.

(2) The team recognises a gap in shared understanding of a requirement, but defers effort in closing this gap to the future. This may be because the gap is out of scope, unimportant at this point in time, or they have run out of time in the meeting.

(3) A gap is detected and effort is made to close the gap by negotiating a new shared interpretation, however agreement cannot be reached during the collaborative episode. This may be due to non-convergence of views and unwillingness to compromise, or it is recognised that it will require an effort too long for the time available in the meeting and so it is postponed.

(4) A gap is detected and effort is made to close the gap by negotiating a new shared interpretation, however information needed to interpret the requirement at the level of detail is not at hand

The model in Figure 2 provides a framework to analyze the potential barriers and enablers of the development of shared understanding from a cognitive perspective. For example, what are the barriers and enabler of establishing the group's attention? With the assumption that attention is a limited cognitive resource and so attention to the shared understanding of a particular requirement is competing with other stimuli, it is important to understand how to minimize the effects of stimuli other than those directing attention related to the requirement in question. Principles from cognition theory also provide explanations for the coordination and propagation of attention in a group that could be useful [15].

This brief explanation and discussion of the empirical cognitive model developed shows its potential for further cognitive analysis

6 Conclusion and Future Work

This paper proposes a novel framework for studying the development of shared understanding of requirements as state changes in requirements-focused Team Mental Models. This, together with ideas from other related cognitive theories, is then used to inform the analysis of field data gathered from an extended, non-participatory observational case study of a team developing software in a commercial setting. This thematic interaction analysis results in a new model of the group-level cognitive tasks that contribute to the collaborative development of shared understanding of requirements. The model is discussed briefly and some new insights into the development of SUR are touched on. A more detailed description of the tasks depicted in the model and their implications for practitioners and researchers will be the subject of a follow-up paper. This will include identifying barriers and enablers of shared understanding of requirements development in light of this mode.

The next stage in the research is to analyze the cognitive model of Figure 2 using Hutchins' Distributed Cognition framework [14] to understand the "how and why" of the framework. Building on the work of [23] this will involve the application of cognitive principles to the model using a modified DiCoT framework as described in [24]. The results of this will then be used to explain the barriers and enablers of SUR development and suggest strategies for better supporting it.

References

1. Kamata, M.I., Tamai, T.: How Does Requirements Quality Relate to Project Success or Failure? Presented at the 15th IEEE International Requirements Engineering Conference RE '07 (2007).
2. Sutcliffe, A.: User-Centred Requirements Engineering. Springer London, London (2002).
3. Bubenko, J.A.: Challenges in requirements engineering. Presented at the Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on (1995).
4. Sadraei, E., aurum, A., Beydoun, G., Paech, B.: A field study of the requirements engineering practice in Australian software industry. Requirements Eng. (2007).
5. Buchan, J., Ekadharmawan, C.H., MacDonell, S.G.: Insights into Domain Knowledge Sharing in Software Development Practice in SMEs. Presented at the APSEC '09: Proceedings of the 2009 16th Asia-Pacific Software Engineering Conference December (2009).
6. Hansen, S., Berente, N., Lyytinen, K.: Requirements in the 21st Century: Current Practice and Emerging Trends. 1–44 (2008).
7. Cannon-Bowers, J.A., Salas, E.: Reflection on shared cognition. Journal of Organizational Behavior. 22, 195–202 (2007).
8. Mohammed, S., Ferzandi, L., Hamilton, K.: Metaphor No More: A 15-Year Review of the Team Mental Model Construct. Journal of Management. 36, 876–910 (2010).

9. Rouse, W.B., Cannon-Bowers, J.A., Salas, E.: The role of mental models in team performance in complex systems. Systems, Man and Cybernetics, IEEE Transactions on. 22, 1296–1308 (1992).
10. Klimoski, R., Mohammed, S.: Team Mental Model: Construct or Metaphor? Journal of Management. 20, 403–437 (1994).
11. Cannon-Bowers, J.A., Salas, E., Converse, S.A.: Shared mental models in expert team decision making. In: Castellan, J. (ed.) Current Issues in Individual and Group Decision Making (1993).
12. Edwards, B.D., Day, E.A., Arthur, W.J., Bell, S.T.: Relationships Among Team Ability Composition, Team Mental Models, and Team Performance. Journal of Applied Psychology. 91, 727–736 (2006).
13. Stahl, G.: Group cognition. The MIT Press, Cambridge, MA (2006).
14. Hutchins, E.: Cognition in the Wild. MIT Press, Cambridge, MA (1995).
15. Heylighen, F., Heath, M., Van, F.: The Emergence of Distributed Cognition: a conceptual framework. Presented at the Collective Intentionality IV (2004).
16. Elo, S., Kyngäs, H.: The qualitative content analysis process. Journal of Advanced Nursing. 62, 107–115 (2008).
17. Jordan, B., Henderson, A.: Interaction analysis: foundations and practice. The Journal of the Learning Sciences. 4, 39–103 (1995).
18. Stahl, G.: How to Study Group Cognition. In: Puntambekar, S., Erkens, G., and Hmelo-Silver, C. (eds.) Computer-Supported Collaborative Learning Series. pp. 107–130–130. Springer US (2011).
19. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering. 14, 131–164 (2008).
20. Yin, R.K.: Case Study Research: Design and Methods. Sage Publications, Thousand Oaks, CA USA (2003).
21. Cañas, A. J., Carff, R., Hill, G., Carvalho, M., Arguedas, M., Eskridge, T. C., et al.: Concept Maps: Integrating Knowledge and Information Visualization. Knowledge and Information Visualization. 205-219 (2005).
22. Cañas, A. J., Hill, G., Carff, R., Suri, N., Lott, J., Gómez, G., et al.: Cmaptools: A Knowledge Modeling and Sharing Environment. Paper presented at The First International. Conference on Concept Mapping Pamplona, Spain. (2004)
23. Sharp, H., Robinson, H.: A Distributed Cognition Account of Mature XP Teams. In: Abrahamsson, P., Marchesi, M., and Succi, G. (eds.) Extreme Programming and Agile Processes in Software Engineering. pp. 1–10. Springer Berlin / Heidelberg (2006).
24. Blandford, A., Furniss, D.: DiCoT: A Methodology for Applying Distributed Cognition to the Design of Teamworking Systems. In: Gilroy, S.W. and Harrison, M.D. (eds.) Interactive Systems. pp. 26–38. Springer Berlin / Heidelberg (2006).