Full citation: Buchan, J., Ekadharmawan, C.H., & MacDonell, S.G. (2009) Insights into domain knowledge sharing in software development practice in SMEs, in Proceedings of the 16th Asia-Pacific Software Engineering Conference (APSEC2009). Penang, Malaysia, IEEE CS Press, pp.93-100.
http://dx.doi.org/10.1109/APSEC.2009.47

Insights into Domain Knowledge Sharing in Software Development Practice in SMEs

Jim Buchan
SERL, Auckland University of Technology
Private Bag 92006, Auckland 1142, New Zealand
jim.buchan@aut.ac.nz

Christian Harsana Ekadharmawan
SERL, Auckland University of Technology
Private Bag 92006, Auckland 1142, New Zealand
christian.harsana@gmail.com

Stephen G. MacDonell
SERL, Auckland University of Technology
Private Bag 92006, Auckland 1142, New Zealand
stephen.macdonell@aut.ac.nz

Abstract

The collaborative development of shared understanding is crucial to the success of software development projects. It is also a challenging and volatile process in practice. Small organizations may be especially vulnerable due to reliance on key individuals and insufficient resource to employ several domain specialists. There is, however, minimal empirical research on sharing domain understanding in the context of small software organizations. In this paper we present the results of a field study of commercial software development practice in which we conducted semi-structured interviews with practitioners from ten such organizations. The study provides insights into practices, perceptions, and challenges related to developing shared domain understanding. Our results show that smaller organizations place particular emphasis on the use of prototypes or existing products to refine and verify domain understanding. Furthermore they perceive the biggest challenge to developing shared understanding as the quality of the client representative(s).

Keywords: software engineering; empirical software; knowledge sharing; requirements engineering

1. INTRODUCTION

Over the last few decades, research has shown that poor quality requirements and poor requirements management practices are among the main factors contributing to software project failure or escalation (see for example [1], [2], [3] and [4]). The high costs of late detection and correction of requirements-related problems, including negative impacts on application maintainability, reliability and usability, are well documented ([5], [6], [7] and [8]). In a similar vein, improvements to requirements engineering (RE) processes and practices have been clearly linked to payoffs in software project productivity, quality and risk management in an empirical study by [9]. The need for continued research into RE process improvement is further emphasised in [10]'s recent summary of the current state of RE research and their proposed research agenda for RE, based on emerging software needs. This paper aims to contribute to the improvement of RE practice by investigating one of the key challenges in RE, namely the development of a shared understanding of the problem (application) domain, which is the foundation on which software solutions are evaluated, designed and implemented.

In the next section we reflect on current thinking with respect to knowledge sharing and RE, in section 3 we set out our research methodology, and in section 4 we report the results of our work and discuss their implications. We briefly conclude the work in section 5 and provide pointers to further work

2. KNOWLEDGE SHARING AND RE

A central contributor to the challenges in RE is the communication and knowledge sharing interactions between a software production (vendor) team and the client stakeholder group in determining the most appropriate set of features and attributes for a new software system. The aim of these knowledge intensive interactions, often embedded in requirements elicitation, analysis and verification activities, is to collaboratively transform the initial uncertain and

ambiguous understanding of the domain problem into an application concept, consistent system requirements, and ultimately a software application that can be used by the target organisation. This involves the client stakeholder group (e.g. users, managers, and domain experts) and the vendor group (e.g. analysts, designers and developers) developing some common level understanding of the problem domain. Although this is most visible at the elicitation effort early in a software project, this process of articulating, sharing, clarifying and sharing understanding is iterative and incremental throughout a software project. At the individual level it involves developing an understanding of the application domain, refining that understanding to a level that is appropriate for the role of that individual and applying it at the time of "need". That (evolving) individual understanding needs to be periodically shared, "tested", verified and agreed upon so that those involved can work cooperatively towards the same goals that create sufficient value for the clients. It is characterised by cognitive, social and organisational interactions that are unpredictable and potentially error-prone. This includes, for example, challenging activities such as developing a shared vocabulary, sharing and internalising both conceptually abstract *and* detailed information about the problem domain, reconciling many points of view from diverse stakeholders, accommodating changing and volatile understanding, as well as periodic verification of some shared representation of the understanding and how it maps to the software solution.

Although it is inherently challenging, the development of shared understanding is critical to the success of a software project. Successful development of a software system is predicated on the vendor team's understanding of the main concepts, goals and purpose of the software system and how well this aligns with a client group's expectations. A number of researchers in RE (see for example [11], [12], [13] and [14]) argue convincingly for the central role domain knowledge sharing plays in RE activities. Empirical evidence from their studies of RE practice demonstrates that high quality requirements are crucially dependent on the client and vendor stakeholders sharing a sufficient level of understanding of the *problem* domain.

The need to improve practice in this area is not lessening either, despite significant advances in modeling, tools and processes over the last few decades. We are seeing the application of software systems to an ever widening diversity of application domain, often conceptually challenging and complex. This broadens and deepens the domain knowledge that developers and other non domain experts have to understand. The need for further research into supporting and comprehending the phenomenon of "developing shared understanding" in practice becomes even more evident in the context of new types of software (e.g. ubiquitous, service oriented, self-managing, or mesh) and emerging development contexts (e.g. global development teams, distributed users, product or market driven development).

Unsurprisingly, there has been considerable research into developing tools, techniques and processes to support the activities and complex interactions that contribute to developing a shared domain understanding in the context of emerging software needs. Many of the findings and proposed approaches, however, are aimed at large organisations, with the tacit assumption that these findings will apply to small organisation (i.e. having less than 50 employees [15]). This point is highlighted in [16], where the authors argue that RE in small organisations is under-represented in research literature. They further observe that such organisations make up a large part of the software industry) and in [17] they estimate that SMEs contribute 80% to economic growth worldwide. Moreover, it is likely that small organisations are more vulnerable to the complexities and volatility of developing a shared domain understanding compared to large organisations. It is widely acknowledged in literature that there are some fundamental operational differences between small and large organisations (see for example the Sept/Oct 2000 issue of *IEEE Software*). In empirical studies of small and medium organisations, [18], [19], and [20] characterise them as having fewer resources to devote to tools and hiring domain experts. Compared to their larger counterparts, small organisations appear to be more concerned about practice rather than "compliance" to formal, defined processes. They also observe that small organisations generally focus on shorter term priorities, which are typically directed towards deliverables. These ideas, strengthened by personal observations of small software companies, suggest that current understanding of and approaches to the development of shared domain understanding may not directly transfer to smaller organisations.

In addition, there is little research that investigates current *practice* for developing a shared domain understanding for software development, especially for small organisations. As pointed out in [16] in their study of RE practice in seven small companies, there is substantial anecdotal evidence (and they present some case-based evidence) to suggest that practitioners in small organisations do not always follow practices described in literature. Knowing what is actually practiced and understanding the challenges, as well as the actual phenomenon, should inform future process improvement in the area of RE and domain understanding for small organisations.

This paper addresses this lack of information on current practices in small organisations in the area of shared domain understanding, and examines the applicability of previously reported findings, generally drawn from experiences with larger organisations, to small companies. In addition, it is the intention of this research to gain insights into practitioners' *perceptions* of their practices and challenges in this area. This is based on a desire to "know" the practitioners more deeply as "customers" of RE research and understand their experiences and needs in this area.

3. RESEARCH DESIGN AND IMPLEMENTATION

3.1 Aims and Methodology

The selection of a research methodology and specific data collection and analysis methods are based on the nature of the research aims and questions. It is the aim of this research to gain insights into the development of shared domain understanding in practice through practitioners' perceptions. In particular, in the context of small software

organisations, the research questions outlined in Table 1 are addressed.

In line with other exploratory studies of this type, a multiple case study method, with semi-structured interviews for data collection, is employed. A semi-structured interview was employed, rather than a formal, structured interview or survey, because it has the advantage of being able to clarify and probe issues and extend the focus of the discussion to interesting aspects *as they arise*. Thus, as observed by [21], a deeper and richer understanding of the phenomenon may be gained. In addition this technique encourages the development of a rapport and trust between the investigator and the interviewee. This is desirable if interviewees are to feel they can freely discuss their practices, challenges and successes.

How do practitioners conceptualise the process of developing a shared domain understanding?
Is the development of a shared understanding important to practitioners?
What practices, techniques and tools do practitioners use to support the development of shared domain understanding? How efficacious are these practices perceived as? How is understanding represented?
Is the development of a shared understanding challenging for practitioners? If so, what are the barriers?
How do the challenges and practices identified for these small organisations compare to existing findings reported in literature?

Table 1. Research Questions

Note that it is not the intention of this initial study to observe practices or analyse artefacts, which are also common sources of data in case studies. In addition, the viewpoint of the study is restricted to the perceptions of the vendor, as represented by senior member of the participating software production teams. Comparison of the viewpoints of representatives of the client stakeholder groups is planned for a future study.

3.2 Case Selection

Invitations to participate in this study were sent to 204 organisations, based on the company size (small), and involvement primarily in software development. We selected candidates who had been operating for at least 5 years to allow for maturing of its practices.

3.3 Research Implementation

Of the candidate organisations invited, 11 organisations initially agreed to participate and 10 organisations actually proceeded with the interviews. Experienced senior-level staff from the organisations were interviewed, with the view that they would have a clear overview of processes as well as some depth of interaction with client representatives, which proved to be the case. Two of the authors of this paper were involved in interviewing all the participants, one facilitating the interview, and the other taking detailed interview notes. The interviews were all recorded on audio tape for later transcription. The interviews were generally located at the place of work, or a neutral venue if requested, and lasted between one and two hours.

The rich and extensive set of data collected from the interviews includes information related to issues such as company size, roles of staff, descriptions of processes, tool support, client communication, knowledge representation, verification of understanding with clients, the quality of client representation, and changes to understanding. Thematic analysis of this data was employed as the method of data analysis. As noted in [22] this is a common method of analysing qualitative interview data in order to identify concepts or themes related to a phenomenon. The main construct being analysed is the *process* of developing shared domain understanding. As [23] points out, analysing a process may provide a holistic view of a system of action, which includes activities, roles, artefacts tools and techniques. The unit of analysis is the (small) organisation rather than specific teams or projects.

4. RESULTS AND DISCUSSION

This section presents the results of the investigation, discusses implications for practitioners, and speculates on some possible directions for addressing some of the issues. After describing the case organisation, this section is structured according to the questions posed in Table 1.

4.1 The Case Organisations

The participating organisations represent a wide diversity of application domains and include 3 product-driven companies and 7 providing bespoke software development services. All of the organisations had been operating for over 8 years and most of them closer to 20 years. The representatives from the companies were all at a senior level ranging from senior systems analyst to company owners, with 8 of them having more than 10 years' experience in the software industry and the other two having extensive business or domain experience.

4.2 The Conceptualisation of Shared Understanding

In order to explore practitioners' views on the processes, challenges and representations of shared understanding, it is firstly important to understand how these practitioners conceptualise the notion of "shared understanding" in their context of software development. This allows for an interpretation of their views informed by their understanding of this concept.

The participants perceive a close relationship between RE and developing shared understanding of the problem domain. They describe the process of sharing understanding in very similar language to RE, describing phases such as "initial elicitation", "knowledge analysis and integration", "artefact development", "validation and verification", and "finalization". In line with contemporary thinking, almost all participants conceptualise domain understanding as evolving and changing over the life of the project and generally considered such changes positive, resulting in "happier clients". Paradoxically, formal "sign off" of requirements is still seen as a required step in confirming agreement of understanding for most participating organisations. This seems to be driven by the contractual nature of the client-vendor relationship, in contrast to the iterative, agile,

partnership model that they describe. The iterations are typically seen as clarification and verification of a larger up-front effort of elicitation and documentation, and may or may not result in changes to understanding and a re-negotiation of requirements.

The main aim of sharing domain understanding is seen as being closely aligned with the development of requirements-related artefacts in the early phases, and verifying and clarifying understanding further into a project. The concept of "requirements" was used quite loosely in general throughout the interviews, with no strong distinctions made between user, business, functional and quality requirements. This lack of clarity could cloud thinking and could result in people talking across purposes or even driving towards a solution with insufficient understanding of the problem.

The distinction between problem space and solution space is an important part of conceptualizing "sharing domain understanding". Interviewees conceptualized this distinction as the delineation of the "business problem" (or "domain problem") and the "software solution", although occasionally this separation was blurred. The interviewees suggested that it is important to demonstrate, to the satisfaction of the client stakeholders and as soon as possible, that the proposed solution or product is a suitable solution for their domain problem.

Another important aspect of conceptualizing "shared understanding" is the role of the different participants in this sharing. The interviewees had a fairly restricted view of the roles of the client stakeholders and software production team and their relationships, as depicted in Figure 1. The knowledge elicitation and sharing between vendor (development) group and client group is perceived as being driven mainly by the vendor group, and is the analyst's role. Typically in a small organization the analyst is the CEO or senior team member since it is perceived as a critical job. This aligns well with the observations of [16] that the Requirements Engineer is typically the CEO or senior sales person. The concept of a "Regulator" captures the need for some of the participating organisations to comply with government or Standards regulations.

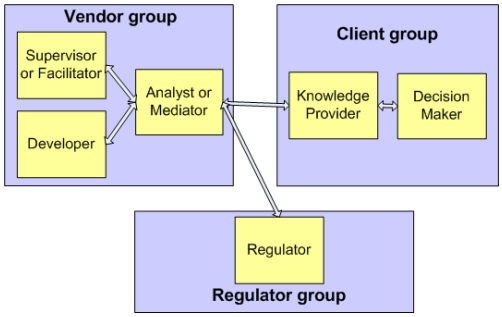

Figure 1. Perceived Stakeholder Roles and Relationships

The consequence of an inadequate level of shared domain understanding was an important aspect of participants' notion on "shared understanding". A lack of shared domain understanding is seen as leading to "miscommunications", "misinterpretations" or "misunderstandings". The consequences of this are described as unexpected client actions, behaviors or decisions, which result in increased frustration levels, increased conflict, inefficient processes, the need for "extra" work, poor quality systems or even project failure.

4.3 The Importance of Sharing Understanding

Gauging the importance of sharing understanding to the participants should provide some indication of the level of effort participants will allocate to improving this shared understanding or related processes. The participants were asked to rank the importance of developing shared understanding on a scale of 1 ("not at all") to 5 ("very important"), and all ten selected the highest score. This seems to reflect more their acknowledgement that a lack of understanding can significantly affect project success, rather than their effort on activities in this area. For example, more than half the participants claim to spend 20-30% of software project effort on activities directly related to sharing domain understanding. They "justified" this "low percentage" (their perception) as stemming from the need to maintain a balance between refining understanding and other development activities such as development, testing and implementation. They described it as a "closed sum" situation where more effort in sharing understanding (and RE in general) would lessen what could be done in the other development activities. This is in conflict with the notion of the high cost of not detecting errors early in the development lifecycle reported in literature and noted by participants in this study.

Participants also believed that their current practices need improvement and that the development of shared understanding is still problematic (as discussed more fully in section 4.4). The conclusion from these findings is that, at least in small organisations, there is the perception that this is an important and (still) challenging area of software development.

4.4 Practice, Tools and Techniques

This research aimed to gain some insights into what practices and supporting tools and techniques the practitioners in SMEs utilize to enhance sharing of domain understanding between the production team and the client stakeholders.

None of the organisations claimed to use a "named" methodology or specialized tool to support the development of shared domain understanding (or even requirements engineering). In fact, quite a diverse set of techniques, tools and practices related to developing domain understanding were reported among the organisations. The results are as summarised, in order of strength of use, in Table 2. Conversational techniques, which ranged from casual conversation to semi-structured interviews, were identified by all ten participants as the most common activity used throughout the life of a project to support shared understanding. Client and vendor stakeholders with strong verbal communications skills are perceived as crucial to the development of shared understanding. This is well supported by evidence from research in [24, 25].

For product-driven organisations the use of regular focus groups is found to supplement informal communications with existing clients to confirm domain understanding with the "market".

Documents are perceived as the "main communication device" for most organisations. This is typically a document related to a specification (e.g. requirements) with both textual and graphical material. The document is seen as "giving people something to talk to". It was widely acknowledged by participants that the utility of the documents by themselves is limited for sharing understanding, being described as "dry", "always ambiguous", "do not help people in reality" and "words are only words". The organisations all stressed the need to complement written understanding with verbal discussion with the client to confirm the "accuracy and meaning" of the documentation. Documents are not seen as a substitute for "getting together with the client as often as we can".

Technique/Tool	Typical functions
Conversational techniques (e.g. conversation, interview)	Eliciting, clarifying & verifying understanding. Conflict resolution.
Prototype/product review	Validating understanding
Document review	Validating understanding
Artefact development tools (e.g. MS Word, Visio, PowerPoint)	Analysing & representing understanding. Artefacts for later verification and communication of understanding.
White boarding session	Triggering new ideas & eliciting understanding
Review of similar products	Triggering new ideas & discussion
Observation	Eliciting understanding
Focus group	Eliciting & verifying understanding
Questionnaire	Eliciting understanding
Iterative User acceptance test	Validating understanding
Issue tracking system	Knowledge storage & management
Glossary	Knowledge sharing

Table 2. Tools and techniques used for developing shared understanding

Face-to-face discussions with client stakeholders is viewed as the most productive mode of communications for sharing understanding, in line with other findings reported in literature (for example [26] and [27]). It was described by participants as improving subsequent communications with clients by getting them "closer" to clients and improving "trust" and "empathy" as well as deepening understanding. A lack of face-to-face communications, in fact, was identified as a serious barrier to developing shared understanding by most of the participating organisations. During later phases of projects, face-to-face meetings were described as being less frequent and needed mainly for issues which were perceived by the vendors as likely to have a significant impact on the project. Phone and email are the prevalent modes of developing shared understanding later in projects. Where clients and developers are a significant distance apart, use of video conferencing (e.g. WebEx) is seen as a viable alternative to face-to-face discussion, although not as "rich".

The use of prototyping techniques, including executables, as well as screen-based and paper-based prototypes, is given particular emphasis by participants as being effective for developing shared domain understanding. It is viewed as a "rich" artefact for validation of understanding of the problem domain and iterative refinement of this understanding. Prototypes are conceived as visual and concrete and allow client stakeholders to test their understanding effectively, particularly since few clients "can assess a concept, they have to see it manufactured." Some participants noted that prototyping predominantly provides feedback on workflow and the user interface, which are not related to solution space rather than the problem domain.

The process of sharing understanding generally involves sharing some artefact that represents the domain understanding that is shared or needs to be shared and verified. The representations of understanding that were viewed as most effective for evolving shared domain knowledge are dominated by informal, loosely structured natural language representations. The challenge of reducing ambiguity inherent in such text was also acknowledged. Participants particularly emphasised the prototype (or product) as a representation of the current state of domain understanding also. Such representations were seen by the vendors as "accessible and understandable" by the clients, as opposed to more formal representations such as UML diagrams, which were seen as unfamiliar to clients. The predominant view they expressed was that most of the shared understanding is in individuals' minds as a result of conversations and informal communications.

In summary, the predominant practices in small organisations seem to rely on largely textual documentation and prototypes as tools for developing and representing and verifying shared understanding. Participants seem to have evolved a diversity of practices that are "good enough" but say they are open to improvement. It appears that there have been few mistakes with an impact serious enough to cause radical change of practices in the participating organisations. Perhaps, as suggested by [16], companies who have experienced such high impact mistakes have already gone out of business.

4.5 Challenges and Barriers

This study sought to explore the challenges and barriers to sharing domain understanding as perceived by the vendor (production) group. These insights should provide directions for future research and identify opportunities for research-practice communications.

All organisations had stories of miscommunications and misunderstandings that had contributed to project difficulties and acknowledged a number of challenges and barriers in developing a shared domain understanding. Table 3 depicts the main challenges identified in order of decreasing strength from this study. Although a number of barriers were identified in this study, this discussion focuses on the two most strongly presented by participants, namely issues with client representation and the differences between the production (vendor) and client groups. It should be emphasised that all the barriers are interrelated both causally and hierarchically and are represented in Table 3 according to the perceptions of the participants of this study.

The challenges presented by poor quality client representatives is consistently emphasised as a significant barrier to shared understanding by all participants. The prevalent view is that the client representative(s) is a key

"interface" to the client organisation and if this relationship and interactions are poor then communications suffers and it is difficult to elicit domain knowledge and verify shared understanding. Characteristics of poor quality client representatives generally related to their perceived lack of knowledge or some form of "resistance" to sharing. Table 4 summarises the main challenges identified by the participants related to client representative quality.

Challenges and Barriers
Inadequate client representatives
Diversity between client and vendor groups
Lack of common language/terminology
Difficult access to key stakeholders
Change in problem understanding
Client uncertainty
Unfamiliar or complex representations
Ambiguity of natural language
Client's internal conflict
Communication timing and frequency
Non-engaging representations
Lack of enough "rich" communication

Table 3. Barriers to Shared Domain Understanding

The most strongly emphasised characteristics of a poor quality client representative relate to perceptions of their domain knowledge. This may be a lack of depth, breadth or inability to clearly articulate the ideas, all resulting in an information need not being satisfied or conceptual understanding being limited. Some representatives were perceived as "poor learners" and slowed the development of new understanding or possible innovation.

Barriers Related to Inadequate Client Representatives
Depth of domain knowledge insufficient.
Unable to articulate tacit domain knowledge.
Is unaware of the views of other stakeholders.
Availability is too restricted. Perceived as inaccessible or lacking commitment.
Actively resistant to participation. (Uncooperative. Unforthcoming. Unwilling to compromise. Always disagrees).
Passively resistant to participation. (Always agrees. Non-committal. Doesn't engage).
Expectations unrealistic. Too demanding
Always changing their mind. Uncertain or inconsistent.
Has a hidden agenda.
Has no authority to make decisions or to speak for other stakeholders.

Table 4. Barriers Related to Client Representatives

Lack of availability of the client representative is also highlighted as being a barrier to developing and sharing domain understanding. A variety of circumstances are identified as being the root cause of this access challenge, including geographical distance, delegation of responsibility, multiple layers of stakeholders, office policy, high cost of representatives, or indifference. There is a strong perception that the representative is often too busy with other work (over-worked perhaps) and consequently takes too long to respond to requests for information or confirmation of understanding, or provides shallow or incorrect responses due to this work pressure.

The other significant area of inadequate client representation relates to the attitudes and behavior of the representatives. Broadly speaking this is perceived as overt uncooperative behavior such as "holding back" feedback or information, or more passive behavior such as always agreeing, with little depth of thought. This was seen as due to a low value being placed on role by the client representative resulting in a lack of "buy-in", "commitment" or "resentment" to the client representation role.

Power is another clear barrier related to effective communication with the client representative. Some are perceived as playing power games with hidden political agendas, so that aspects of understanding were withheld to the advantage of the client in some way. Related to this is the frustration expressed by some participants when trying to negotiate understanding and perhaps reach a compromise or decide on alternative views, when the representative doesn't have the authority to speak for the organisation and must consult with a higher authority.

It is clear from the interviews that most participants placed significant reliance on getting quality domain expertise and quality feedback and verification of understanding from the client representatives. They generally perceived the selection of the client representative(s) as largely out of their control, although two organisations report influencing the selection of the customer representative through negotiation. Another three organisations reported employing their own domain experts, who act as "proxy" clients. Presumably the client organisations don't actively select a poor quality representative for a project, so the question remains as to how this situation arises? No clear causes are offered by the participants, although the "blame" was certainly placed with the client group by the (vendor) interviewees. While the quality of client representation is discussed in RE literature ([26], [28]) it appears that for small organisations it is perceived as a particularly significant and frequent barrier to developing shared understanding. Perhaps this is because larger teams in larger organisations have multiple points of contact. This could be a fruitful area for process improvement and better tool support. How can a more visible "client management" process be designed that will promote the selection of the client representatives based on appropriate criteria and support them to engage and commit, despite having competing work pressures?

"Diversity" between the client and vendor groups is also identified strongly as a barrier to shared understanding. This is described by the participants as being linked to the differences in individuals' characteristics such as their experiences, depth of knowledge, abilities to conceptualise, values, risk tolerance, and priorities. It is conceptualised as resulting in difference "trends" between the groups that develop over a series of inter-group interactions. This can lead to increased misunderstandings, miscommunications or misinterpretations that disrupt clear communications and hinder the development of shared understanding. Particularly noted by participants are the differences in depth of knowledge between the two groups, more technical on the vendor side and business oriented in the case of the client group. Three particular issues that relate to sharing

understanding, which are a result of this diversity, are emphasised by participants. One relates to the "difficulty to get them to speak the same language". This is seen as a significant barrier to developers gaining a sufficient understanding of the business processes and goals, and being the root cause of "some projects failing badly". Another barrier, related to this, is the problem of the developers "jumping into the coding process before they understand the business goals and processes", resulting in a "cycle of change, change, change", which is problematic to accommodate. Another interesting challenge relates to the lack of "big-picture" some users exhibit. This is seen as resulting in users often getting "lost" and "missing the point because they don't understand what the business is trying to do". This may be in part due to the tendency for the production group to move from domain problem understanding to software solution too soon.

A commonly cited issue with requirements management is the difficulty in managing changes to scope and requirements. When prompted about changes to domain understanding, participants expressed the view that, although they considered changes "normal", it can be problematic if not "well managed". They described challenging experiences such as overly frequent changes, clients not sharing changes with the vendor, and lack of clarity on the wider impact of new understanding.

Overall, participants identified a broad range of interrelated barriers and challenges to adequate sharing of domain understanding. The potential for client representatives to inhibit the sharing of domain understanding between the two groups was emphasised in the frequency and strength with which interviewees, unprompted, raised this as an issue. The next most forcefully expressed barrier related to the diversity in "world views" and experiences of the vendor and client groups. This contributed to a number of communications issues that obstructed sharing understanding between the two groups.

5. CONCLUSION AND FUTURE WORK

This study found evidence that shared problem understanding is perceived as being an important but challenging contributor to software quality in small organisations. This suggests that practitioners in small organisations may be open to suggestion for process improvement and technology transfer from research to practice may be less of a challenge than with larger organisations.

It was also discovered that the notion of shared domain understanding is conceptualised as being closely related to activities in requirements engineering and that it is thought of as a complex, iterative, and evolutionary process. The wide diversity of practices used by these organisations possibly reflects the diversity of both experiences and market niches they occupy. The absence of formal methodologies to support RE activities, while on the face of it could be interpreted as a research-practice gap, is more likely to reflect the sufficiency of practices that are "good enough". If researchers are to influence SMEs to improve processes or supporting tools in this area, then the value of such a change must be clearly evidenced and communicated to practitioners. The demonstrated business value of proposed improvements must be such that they override the (perceived) overhead of going further than the sufficiency of current processes.

Lack of distinctions between the problem and solution space and different types of requirements became apparent throughout the interviews and this may cloud some of the understanding in this area. Perhaps distinguishing these concepts more carefully in literature, with clearer justifications will assist this knowledge transfer.

Twelve barriers to shared domain understanding are identified in this study. While these barriers generally align with literature, a greater emphasis was given to the quality of the client representative(s). To some extent this may reflect the reliance small organisations place on the domain expertise of the client organisation. A better understanding of client representative selection and management may suggest techniques which could increase the "visibility" of this issue and support the management of client representative selection, commitment and communication. These findings suggest that such improvements could have a significant impact on domain knowledge sharing in small organisations.

Another significant finding is the (over) reliance on prototype techniques as a representation of "embedded" domain understanding and its frequent mention as method of *early* verification of this understanding. This may lead to the selection of a solution system approach before sufficient shared understanding of the problem domain is achieved. A consequence of this could be that possible alternative domain solutions are not considered or a solution is developed for the "wrong" problem. A technique that models the domain understanding *at a conceptual level* and is shareable, easily manipulated as well as (cognitively) accessible to both client and vendor stakeholders, could complement the more concrete representations such as a prototype. This would encourage deeper domain understanding and verification at a more conceptual level, and having the "big picture" front of mind when appropriate.

It was beyond the scope of this study to investigate the development of shared domain understanding from a *client* stakeholder perspective, but this would be an interesting comparison that could provide some useful insights for software development practitioners and researchers.

6. REFERENCES

[1] F. P. Brooks, *The Mythical Man-Month*, Anniversary ed.: Addison-Wesley, 1995.

[2] M. I. Kamata and T. Tamai, "How Does Requirements Quality Relate to Project Success or Failure?," in *Requirements Engineering Conference, 2007. RE '07. 15th IEEE International*, 2007, pp. 69-78.

[3] M. Keil, J. Mann, and A. Rai, "Why software projects escalate: an emperical analysis and test of four theoretical models," *MIS Quarterly*, vol. 24, pp. 631-664, 2000.

[4] U. K. Kudikyala and R. B. Vaughn, "Software requirement understanding using pathfinder networks: Discovering and evaluating mental models," *Journal of Systems and Software*, vol. 74, pp. 101-108, 2005.

[5] D. J. Flynn, *Information systems requirements: determination and analysis*: McGraw-Hill, London, 1992.

[6] L. Hoffmann and F. Lehner, "Requirements Engineering as a Success Factor in software projects," *IEEE Software*, vol. 18, pp. 58-66, 2001.

[7] B. Boehm, *Software Engineering Economics*. Upper Saddle River, NJ: Prentice Hall, 1981.

[8] A. Katasonov and M. Sakkinen, "Requirements quality control: a unifying framework," *Requirements Engineering*, vol. 11, pp. 42-57, 2006.

[9] D. Damian and J. Chisan, "An empirical study of the complex relationship between the requirements engineering process and other processes that lead to payoffs in productivity, quality and risk management," *IEEE Transactions on Software Engineering*, vol. 32, pp. 433-453, 2006.

[10] B. Cheng and J. Atlee, "Research Directions in Requirements Engineering," in *International Conference on Software Engineering, Future of Software Engineering*, 2007, pp. 285-303.

[11] E. G. Alcázar and A. Monzón, "A process framework for requirements analysis specification," in *Proceedings of 4th International Conference on Requirements Engineering*, 2000, pp. 27-35.

[12] R. Offen, "Domain Understanding is the Key to Successful System Development," *Requirements Engineering*, vol. 7, pp. 172-175, 2002.

[13] A. Osada, D. Ozawa, H. Kaiya, and K. Kaijiri, "The role of domain knowledge representation in requirements elicitation," in *25th IASTED International Multi-Conference Software Engineering*, Innsbruck, Austria, 2007, pp. 84-92.

[14] J. Coughlan, M. Lycett, and R. D. Macredie, "Communication issues in requirements elicitation: a content analysis of stakeholder experiences," *Information and Software Technology*, vol. 45, pp. 525-537, 2003.

[15] EU, "The new SME definition. User guide and model declaration. ," European Commission, 2005, http://www.ec.europa.eu/enterprise/enterprise_policy/sme_definition/sme_user_guide.pdf.

[16] J. Aranda, S. Easterbrook, and G. Wilson, "Requirements in the wild: How small companies do it," in *Proc. 15th IEEE International Requirements Engineering Conference RE '07*, 2007, pp. 39--48.

[17] S. Pavic, S. C. L. Koh, M. Simpson, and J. Padmore, "Could e-business create a competitive advantage in UK SMEs?," *Benchmarking: An International Journa*, vol. 14, pp. 320-351, 2007.

[18] E. Kamsties, K. Hörmann, and M. Schlich, "Requirements engineering in small and medium enterprises," *Requirements Engineering*, vol. 3, pp. 84-90, 1998.

[19] F. Pino, F. García, and M. Piattini, "Software process improvement in small and medium software enterprises: a systematic review," *Software Quality Journal*, vol. 16, pp. 237-261, 2008.

[20] A. Atherton, "The uncertainty of knowing: An analysis of the nature of knowledge in a small business context," *Human Relations*, vol. 56, pp. 1379-1398, 2003.

[21] L. Karlsson, A. G. Dahlstedt, B. Regnell, J. N. Dag, and A. Persson, " Requirements engineering challenges in market-driven software development - An interview study with practitioners.," *Information and Software Technology*, vol. 49, pp. 588-604, 2007.

[22] M. Miles and A. Huberman, *Qualitative Data Analysis*. Sage, CA: Thousand Oaks 1994.

[23] W. Tellis, "Application of a case study methodology," *The Qualitative Report*, vol. 3, 1997.

[24] B. Curtis, H. Krasner, and N. Iscoe, "A field study of the software design process for large systems," *Communications of the ACM*, vol. 31, pp. 1268-1286, 1988.

[25] K. E. Emam and N. H. Madhavji, "A field study of requirements engineering practices in information systems development," in *Proceedings of the Second IEEE International Symposium on Requirements Engineering*, 1995, pp. 68-80.

[26] L. Cao, & Ramesh, B., "Agile requirements engineering practices: An empirical study," *IEEE Software*, vol. 25, pp. 60-67, 2008.

[27] D. Damian, "Stakeholders in global requirements engineering: Lessons learned from practice," *IEEE Software*, vol. 24, pp. 21-27, 2007.

[28] C. Lu, Chu, W. C., C. Chang, and C. Wang, "A model-based object-oriented approach to requirements engineering (MORE)," in *31st Annual International Computer Software and Applications Conference*, 2007, pp. 153-156.