TACIT KNOWLEDGE EXTRACTION FOR SOFTWARE REQUIREMENT SPECIFICATION

Serbülent Ünsal
Akgun Software A.Ş.
Akgun Sofrware A.Ş. Çetin Emeç Bulvarı 4.Cad. Burak Apt No:5 D:3 06460 A.Öveçler, ANKARA/TURKEY
serbulentu@gmail.com

Meryem Ayas
Havelsan A.Ş.
Havelsan A.Ş. Ar-Ge Binası, ODTU Teknokent Yerleşkesi, ANKARA/TURKEY
meryemayas@hotmail.com

Tunç Durmuş Medeni
TURKSAT A.Ş.
Konya Yolu 40.km, Gölbaşı/ANKARA/TURKEY
tuncmedeni@gmail.com

Abstract

Knowledge extraction is becoming an important research area for the organizations in order to get and share the knowledge. Most important and useful part of the knowledge extraction is dedicated to tacit knowledge extraction. There are already known methods to acquire the tacit knowledge. Yet, these methods are mostly general approaches applicable to all the areas in need of tacit knowledge extraction and become too abstract when applied to a specific domain. One such specific domain is the requirement specification process for the software project development. Our own experiences in the area as well as the scientific researches have shown that Software Requirement Specification (SRS) process has field-specific problems that need to be eliminated by using the suitable tacit knowledge extraction techniques. For example, the experts and/or users may not have a clear idea of their requirements. They may also be technically unsophisticated or have different vocabularies than the software developers. Benefiting from the existing body of academic literature in the related fields, as well as co-authors' experience from their domains of practice, this paper aims to find the concrete methods for extracting the tacit knowledge in the area of software project development with specific implications for these academic fields and practice domains, as well as more general suggestions for all related or concerned.

Keywords: *Tacit Knowledge extraction, Expert Knowledge, Software Requirement Specification (SRS), Knowledge Engineer (KE)*

JEL Classification: O30

1 INTRODUCTION

Tacit knowledge is "a non-linguistic, non-numerical form of knowledge that is highly personal and context specific and deeply rooted in individual experiences, ideas, values and emotions" (Gourlay, 2002:p.2). It is most commonly referred to be -what people called- "know-how" or

"experience" which makes it hard to verbalize, elicit, write down or extract. On the other hand, tacit knowledge is one of the top demands of the organizations sought for when they hire people or maintain the current projects. Therefore, the extraction of tacit knowledge is an important field not only for the academic purposes but also for the organizational needs and especially becomes more challenging when the software requirement specification (SRS) is in question. Co-authors' different experiences in various domains of the Turkish ICT sector confirms the challenging nature of expert knowledge extraction for SRS. Accordingly, with this work we aim to shed light on extraction of experts' tacit knowledge for software requirement specification.

With respect to this aim, the content of the paper is organized as follows: Firstly, we make a brief introduction to the general tacit knowledge extraction methods/approaches that are applicable to the software requirement specification area. Shortly after, we identify the extraction problems specific to the software requirement specification by dividing the problems into two categories, namely User/Expert Based Problems and Engineer/Developer Based Problems. Then we provide a deeper analysis on each problem and suggest new approaches as solutions while finding also the general methods that are applicable to solve each problem. We will conclude with an outlook of future work and further research opportunities.

2 EXTRACTION METHODS APPLICABLE TO SRS PROCESS

Extraction of (experts') tacit knowledge is an issue discussed in the literature. In general terms, Nonaka & Takeuchi (1995) suggest to use metaphors, analogies and models to externalize tacit knowledge into explicit knowledge, distinguishing these two types of knowledge and four types of knowledge conversions (Socialization, Externalization, Combination and Internalization)

To address the challenging nature of tacit knowledge extraction, also a number of methods have been developed so far. Yet the problem is there are not many studies to distinguish between these methods so that they make easier to resolve the area specific problems experienced while a tacit knowledge extraction is in place. Each problem may require a different extraction technique to be applied. Some of these techniques are briefly described in this paper.

2.1 Interviews

An interview is a face to face discussion between the Subject Matter Experts (SMEs) who posses the domain knowledge and Knowledge Engineers (KEs) who ask questions and/or observe the expert solving problems. Interviews are simply divided into three categories which are

- Unstructured Interviews:

  This type of interview is also known as the kick-off interview. It has a rough agenda and there is no pre-defined structure of the meeting. It acts as an ice-breaker between the expert/user and the engineer and gives them the flexibility to explore the domain. Yet, it is inefficient to gather detailed knowledge in this type of interviews.

- Semi-Structured Interviews:

  This interview has a highly structured agenda which makes the KEs prepare the questions before the interview and makes the expert prepare the responses. Although the focus is on the key questions it has the flexibility to ask subsequent questions. It is also known as general knowledge gathering session.

- Structured Interviews:

  Structured Interviews are not flexible to the engineer whose questions are all pre-established before the interview. It often involves filling in a matrix or other diagramming notations. It is also known as specific knowledge gathering sessions

A basic one-to-one interviewing technique, Output-Input-Middle method, is also suggested to be useful for extracting expert knowledge by KE.

- Output: Identify the answers or solutions to the problem under discussion (goals), focusing on understanding subtle differences between goals
- Input: Identify the sources of information that the expert uses to deduce the solution/answer, making sure how these inputs are identified, determined, or generated is known and understood
- Middle: Determine the links between the inputs and outputs that represent the core of the expert's knowledge (Some inputs may not be required initially, but may be requested later after the initial inputs are interpreted and Intermediate goals/hypotheses may be required to complete the connections)

To complement interviews, other elicitation techniques such as observational elicitation and role reversal can also be used (ibid), or Capability Review Sessions and commentary/think aloud problem-solving techniques can be incorporated, when appropriate. Sometimes, the recordings of the interactions between SME and KE could be handy, or the facilitation of a mediator between SME and KE could also be necessary.

2.2 20 Questions Technique, Teach-back

In 20 questions technique, user/domain expert asks 20 (Yes/No) questions to engineer about the project. The questions and the order give the relationships and the priority of user requests.

Teach-back method also encourages the KEs to describe what they learned from the previous knowledge gathering sessions and let the SMEs comment on the KEs descriptions by correcting the misunderstandings. Although the method is weak when used alone, it may be a supportive technique when used with other extraction methods.

2.3 Card Sorting

The card sorting method is used to generate information about the associations and grouping of specific data items. Participants in a card sort are asked to organize individual, unsorted items into groups and may, depending on the technique, also provide labels for these groups. In a user-centered design process, it is commonly used when developing site architecture but has also been applied to developing workflows, menus, toolbars, and other elements of system design.

- Card sorting may be conducted as a low tech method using index cards or post-it notes, or may be automated using one of several software packages
- Card sorting may be conducted as a series of individual exercises, as a concurrent activity of a small group, or as a hybrid approach where individual activity is followed by group discussion of individual differences.

- Card sorting is usually conducted as a specific activity in the early design phase of a project for defining an architecture, but can similarly be used during a product evaluation to determine if usability issues are due to problems with grouping or group labels

Sorting and grouping have long been studied within psychology and the research dates back at least to the 1950s. Numerous, non-peer reviewed descriptions, case studies, and blogs have been written in the last several years on the technique and its use in the user-centered design process, but only a few peer reviewed articles on the technique have been published and little is known of its validity or reliability as a means of directly producing a useful and usable architecture. Instead, card sorts are generally used to provide insight that is used by a practitioner to generate an architecture.

Figure-1: A Sample Card Sorting Practice

Source: Gravity Freedom (2007)

2.4 E-mail, E-Discussion Boards

As a simple tool for asynchronous communication, e-mail is ranked as an effective and most important method in terms of knowledge conversion from one to another. (Harris, 2008) It is also identified as a topmost socialization tool for extracting and sharing unstructured knowledge. (Dfouni and Croteau, 2004) Email is a preferred since it reduces the time wasted during the interviews and gives the both sides to get prepared before the meetings.

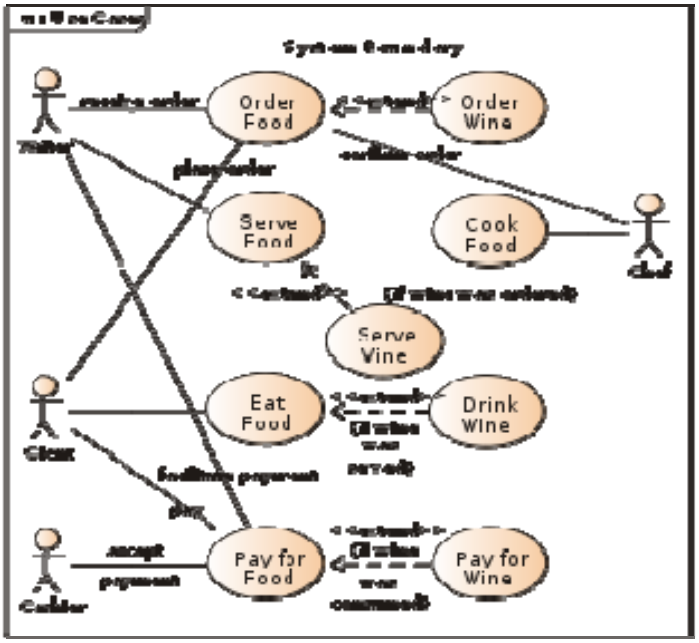
E-Discussion boards are known as asynchronous communication tools which allow its members to post messages, ask or answer the questions online. E-Discussion boards are found to be useful for sharing beliefs and mental models of individuals. (Dfouni and Croteau, 2004) They also refer to bulletin boards or message boards typically.

2.5 Use Case Diagrams

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show

what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Figure-2: A sample use case diagram



Source: Wikipedia (2011)

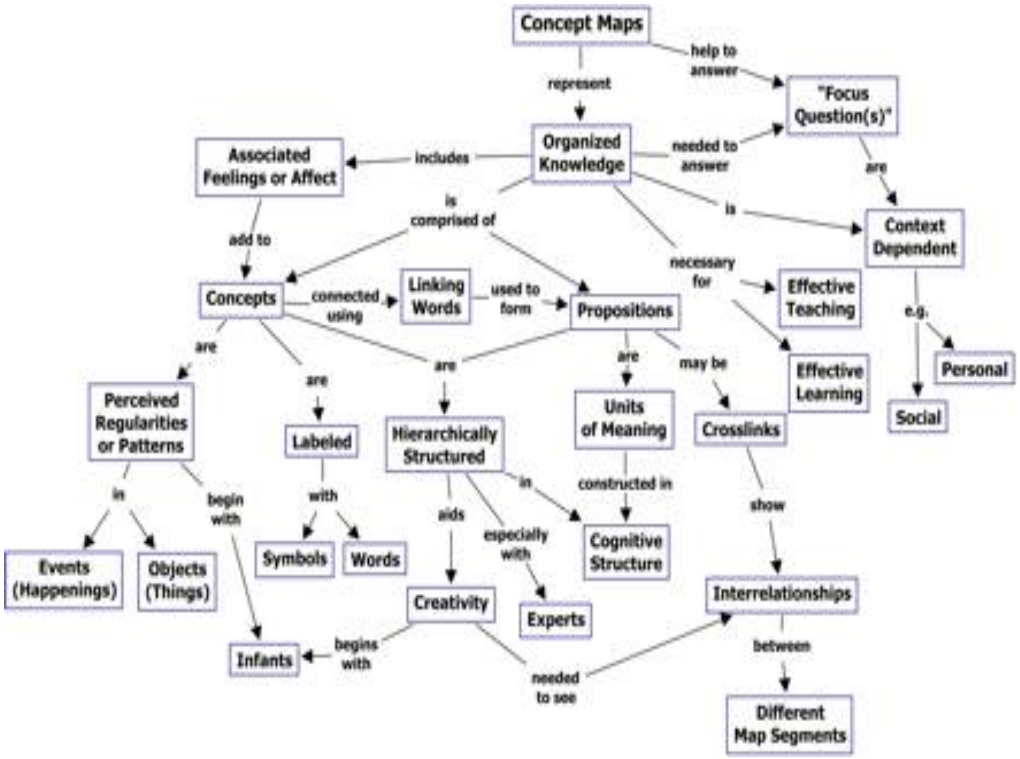
2.6 Concept Maps

Concept maps are graphical tools for organizing and representing knowledge. They include concepts, usually enclosed in circles or boxes of some type, and relationships between concepts indicated by a connecting line linking two concepts. Words on the line referred to as linking words or linking phrases, specify the relationship between the two concepts.

We define concept as a perceived regularity in events or objects, or records of events or objects, designated by a label. The label for most concepts is a word, although sometimes we use symbols such as + or %, and sometimes more than one word is used. Propositions are statements about some object or event in the universe, either naturally occurring or constructed. Propositions contain two or more concepts connected using linking words or phrases to form a meaningful statement.

Sometimes these are called semantic units, or units of meaning. Figure shows an example of a concept map that describes the structure of concept maps and illustrates the above characteristics. (Novak & Cañas 2006)

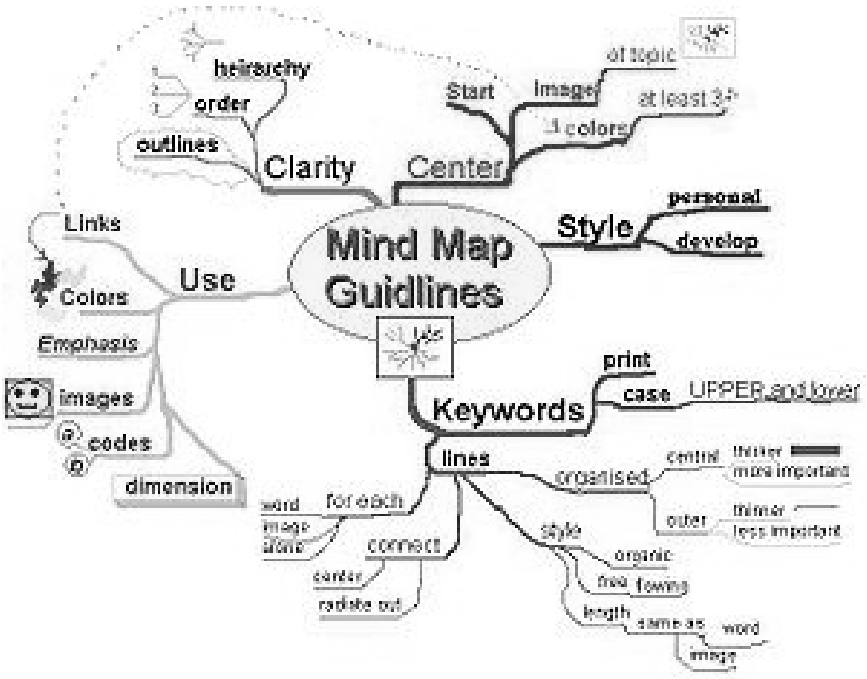
Figure-3: A sample concept map



Source: Institute of Human and Machine Cognition (2008)

2.7 Mind Maps

Mind mapping is a quite novel technique for information visualizing. Put more simply, cognitive maps are a method we use to construct and accumulate spatial knowledge, allowing the "mind's eye" to visualize images in order to reduce cognitive load, and enhance recall and learning of information. Also another type called fuzzy mind maps are available. In this type mind maps have weights between relationships. These weights represent power of the relationship.



Source:Study Habits (2011)

2.8 Matrix Based Techniques, Repertory Grids

In this technique concepts are written in 2 dimensional matrices like Concept vs. Properties, Problems vs. Solutions, Tasks vs. Resources, Hypothesis vs. Diagnosis. Thus all components of the project could be seen comparatively.

A repertory grid is also a list of specific characteristics of a domain that are to be evaluated by an expert

- Mathematically: an attribute-value vector,
- Attributes are also sometimes called elements or labels,
- Values can be binary or a range of values,
- A construct is an attribute-value pair (along with the specification of the range, i.e., set of allowed values)

Automated tools exploit the idea of repertory grids by trying to help elicit what attributes are important for the domain, and what range of values the attributes should have.

Figure-4: A Sample Repertory Grid

ELEMENTS 10, CONSTRUCTS 14, RANGE 1–5
PURPOSE: Staff appraisal

	Staff member No.	1	2	3	4	5	6	7	8	9	10	
1	Intelligent	1	1	4	5	3	3	5	2	3	5	Dim
2	Willing	1	2	4	5	1	1	4	3	1	2	Unwilling
3	New boy	1	2	3	5	4	4	4	1	4	3	Old sweats
4	Little supervision	3	1	4	5	2	1	5	2	2	3	Needs supervis.
5	Motivated	1	1	4	5	2	2	5	3	3	2	Unmotivated
6	Reliable	3	2	2	5	1	1	5	1	2	3	Unreliable
7	Mild	3	4	5	2	2	3	1	5	4	5	Abrasive
8	Idea person	1	1	5	4	2	3	1	3	4	4	Staid
9	Self-starter	2	1	5	5	1	3	5	3	4	5	Needs a push
10	Creative	1	1	5	5	2	3	4	3	4	5	Uncreative
11	Helpful	4	3	4	2	3	5	1	4	5	5	Unhelpful
12	Professional	1	2	3	3	2	1	5	2	4	4	Unprofessional
13	overall rating high	2	1	3	4	1	2	5	2	3	4	overall rating lo
14	Messer	2	2	5	4	3	5	1	5	3	1	Tidy
	Staff member No.	1	2	3	4	5	6	7	8	9	10	

Source: Becerra-Fernandez, et al., Dekai Wu (2004 & 2007)

2.9 Learning Laboratory, Automatic Knowledge Extraction

Learning laboratories are simulation environments. The main aim of these facilities is educating junior staff with observation of experts. But we claim that these facilities also could be useful for extraction of tacit knowledge. Since real life processes could be easily observed and recorded in this facilities extraction of tacit knowledge could be possible for a requirement analysis.

Automatic knowledge extraction is the process of observing a domain expert when he performs an action and extract the tacit knowledge. An example could be an operator who fixes errors with a computer interface on a production band. When a machine stops s/he finds the error, calibrate the production robot and start system again. All these steps could be recorded with an intelligent system and tacit knowledge of the operator can be converted to rules or cases to be used and updated in future times. Thus tacit knowledge of the operator could be converted to explicit knowledge.

There are various other simple of sophisticated methods or techniques that can be directly or indirectly applied for knowledge extraction, However, after this provision of selected general knowledge extraction methods, we can now analyze and discuss how we can address tacit knowledge extraction problems specific to SRS process. The analysis has benefited from our experience and review of literature, as well as participant practitioner feedbacks from a two-week seminar in Knowledge Management and Technologies Graduate Class in 2010 in Informatics Institute, METU, Turkey.

3 TACIT KNOWLEDGE EXTRACTION PROBLEMS SPECIFIC TO SRS PROCESS

Problems can be analyzed under two main categories which are User/Expert Based Problems and Engineer/Developer Based Problems. User Based problems are the problems faced due to the limitations or incapability of the users/experts. These problems can be viewed as:

- The users don't have a clear idea of their requirements,
- Some users are technically unsophisticated.
- Communication with the users is slow.
- Users often do not participate in reviews.

On the other hand, there are problems related to Engineers/ Developers which makes the SRS Process inefficient or hard to complete. These kinds of problems can be viewed as:

- Technical personnel and end-users may have different vocabularies. Consequently, they may wrongly believe they are in perfect agreement.
- Engineers and developers may try to make the requirements fit an existing system or model, rather than develop a system specific to the needs of the client.
- Analysis may often be carried out by engineers or programmers, rather than personnel with the people skills and the domain knowledge to understand a client's needs properly.

The problems and suggestions addressing those problems are discussed in detail, as below.

3.1 User/Expert Based Problems

3.1.1 The users don't have a clear idea of their requirements

A common problem with the users is the degree of their awareness of what they want. Most of the users do not know what they really need and cannot express their thoughts. Sometimes, they do not even know that they know. In such cases, more visualization and face to face sessions are needed. In order to overcome this problem interview method must be applied, supported with the visual methods such as Concept Maps, Mind Maps and Use Case Diagrams. In the first meeting (the Kick-off Meeting) the expert(s) and the engineer(s) will have the chance to explore the domain and they will establish some ground rules and mutual understanding of expectations. Putting a Capability Review Session between the first meeting and the second meeting might be a good idea in order to make the expert to understand the limitations and capabilities of the system. This capability review session should be supported by use case diagrams and other diagram/visualization based techniques to make the users have more concrete understanding of the requirements and the system.

3.1.2 Some users are technically unsophisticated

Technically unsophisticated users may be a nightmare for the developers/engineers in some cases. This causes both sides not to understand each other and waste time having arguments. Since the user is unable to understand what can/ cannot be done, they may be requesting impossible, or hard and too late to develop characteristics or requirements. In this case, having a “mediator” who will act as an interface or bridge between the two sides is a must. The mediator will find a way to establish or improve the dialog between the parties. The efficiency of the agreement depends on

the skills and the experience of the mediator. However, having a mediator only will not totally address the problem and a technical review between the meetings might be required to make the user familiar with the technical side of the task.

3.1.3 Communication with the users is slow

Another common problem is the rate of the communication between two sides. Sometimes, especially in large-scale projects, a variety of meetings might be needed. This may be either due to the scale of the modules to be covered or due to the unresolved parts in the previous meetings. In order to handle this problem e-mailing will be good solution. E-mail makes it easier to understand the unclear parts, to get prepared before coming to the meeting and to have an individual time to work on the parts and respond to the others.

3.1.4 Users do not participate in reviews

Not having all the necessary people in the meetings is an important problem for SRS process. The experts are mostly busy people and they usually have too many meetings to join. This will end up with the absence of the expert in the meeting and will probably cause to arrange some other meeting in some other time, wasting other participants' time. Another obstacle for the expert to join the meeting may be developer/hired company may be in a different city. In both cases having the required people might not be possible. To resolve this problem E-mailing, E-Discussion Boards, and/or Tele/Video Conferencing need to be used. Using any of these methods will reduce time wasted by the participants.

3.2 Engineer/Developer Based Problems

3.2.1 Technical personnel and end-users may have different vocabularies

It is a common problem that users and technicians using different languages. In most of the cases, users don't understand technicians' vocabulary and technicians don't know users' domain. There are some methods to solve this issue. First of all a mediator who have information about both sides would be very effective. Also in semi structured interviews, technical team and users learn each other's languages. For a productive interview, technical team should make a domain research before meeting. The main problem of research is distinguishing which term/knowledge is important and which is not in the domain. We suggest using tag clouds ref as a special mapping/visualization tool to handle this issue. A tag cloud of the domain will represent important terms of the domain.

Collaborative mind mapping would be another method for mediation of user and technician. When they try to create a mind map together they will understand each other's point of view. Teach-back methodology is our final suggestion for this problem. In this method domain expert (end-user) comments on what the engineer describes about the learning of previous sessions. Thus engineer could correct misunderstandings.

3.2.2 Engineers and developers may tend to fit to an existing model rather than a new specific product.

To summarize this problem, "if you have a hammer everything starts to seem like a nail" could be said. Sometimes assimilating one problem to another solved one might be helpful. But if they are not really similar, lots of time will be wasted for wrong analysis. Commentary/think aloud problem-solving technique could be a solution for this case. When engineer thinks aloud

user/domain expert could correct his/her assumption about the problem. One drawback of this solution is cognitive overload on user. To overcome, off-line reporting using a recordable media (video, etc.) could be used.

Also in a learning laboratory engineers can share experience of users, this experiences may guide to custom-made solutions. Like learning laboratory, observation technique is used to understand real process in domain. In observation technique engineer observes the expert and take notes while they perform the tasks. This technique could also be supported with video recordings. Finally 20 Questions technique could be useful to design a product that really fits to user requests.

4 CONCLUSION

The content of the paper has been organized as follows: First, we have made a brief introduction to the general tacit knowledge extraction methods/approaches that are applicable to the software requirement specification area. Then, we have identified the extraction problems specific to the software requirement specification by dividing the problems into two categories, namely User/Expert Based Problems and Engineer/Developer Based Problems, providing a deeper analysis on and suggest solutions to each problem.

While we have tried to identify new approaches as solutions to existing problems, we have also found useful certain general methods as applicable to solve specific problems in different domains. However, these methods and problems are not mutually exclusive, or suggested methods and experienced problems do not necessarily match. It should then be acknowledged that sometimes just common sense and simple tools could work, and it is always good to allow certain buffer space and time for sorting out misunderstandings and conflicts that could arise between SMEs and KEs, among whom committed and trusted interaction is important. Accordingly, most of the discussed methods and problems, as well as other new ones can also be found in different domains' of literature, which can be revealed by future research on reviewing academic and practitioner literature.

We should also note that this paper mainly reflects co-authors' subjective experience and analysis. We aim to apply the results of this paper to our own practice. We also believe these results and findings would be useful for others that study practitioner experience or academic literature on tacit knowledge extraction of experts for SRS or related topics.

BIBLIOGRAPHY

Gourlay, Stephen (2002), "Tacit knowledge, tacit knowing or behaving?", Kingston Business School, Kingston upon Thames,UK

Nonaka, Ikujiro; Takeuchi, Hirotaka (1995), The knowledge creating company: how Japanese companies create the dynamics of innovation, New York: Oxford University Press

Tagger, Ben, An Enquiry into the Extraction of Tacit Knowledge,

http://www.cs.ucl.ac.uk/staff/B.Tagger/FinalIMR.pdf [Accessed 7.01.2011]

Wikipedia (2011), Requirement Analysis,

http://en.wikipedia.org/wiki/Requirements_analysis [Accessed 7.01.2011]

Epistemics (2003), Knowledge Acquisition,

http://www.epistemics.co.uk/Notes/63-0-0.htm [Accessed 7.01.2011]

Irick, Michael L. (2007), "Managing Tacit Knowledge In Organizations", Journal of Knowledge Management Practice, Vol. 8, No. 3, pp.1-5.

Memon, Nasrullah , Knowledge Elicitation – Converting Tacit Knowledge to Explicit,

http://www.mip.sdu.dk/~memon/10-2.pdf [Accessed 7.01.2011]

Usability Body of Knowledge (2007), Card Sorting,

http://www.usabilitybok.org/methods/card-sorting [Accessed 12.02.2011]

Wikipedia (2011),Use case diagram,

http://en.wikipedia.org/wiki/Use_case_diagram [Accessed 07.01.2011]

Institute for Human and Machine Cognition (2006), The Theory Underlying Concept Maps and How to Construct and Use Them,

http://cmap.ihmc.us/publications/researchpapers/theorycmaps/theoryunderlyingconceptmaps.htm [Accessed 23.02.2011]

Becerra-Fernandez, et al (2004), Dekai Wu (2007), Knowledge Management

www.cs.ust.hk/~dekai/600G/notes/KM_Slides_Ch10.pdf [Accessed 12.02.2011]

Gravity Freedom (2011), Card Sorting Your Way to an Improved User Experience

http://gravityfreedom.com/2007/06/14/card-sorting-your-way-to-an-improved-user-experience/ [Accessed 12.02.2011]

Study Habits (2011), How To Mind Map

http://www.study-habits.com/how-to-mind-map [Accessed 12.02.2011]