Journal of Universal Knowledge Management, vol. 1, no. 2 (2006), 59-68
submitted: 18/10/06, accepted: 24/10/06, appeared: 28/10/06 © J.UKM

Learning Software Organisations and Requirements Engineering: First International Workshop (LSO+RE 2006)

J.UKM Special Issue

Andreas Birk[1], **Torgeir Dingsøyr**[2,3], **Stefanie N. Lindstaedt**[4], **Kurt Schneider**[5]
[1]sd&m AG, software design & management
Löffelstraße 46, D-70597 Stuttgart, Germany
andreas.birk@sdm.de
[2]Department of Computer and Information Science
Norwegian University of Science and Technology
NO-7491 Trondheim, Norway
[3]SINTEF Information and Communication Technology
NO-7465 Trondheim, Norway
torgeir.dingsoyr@sintef.no
[4]Know-Center
Inffeldgasse 21a, 8010 Graz, Austria
slind@know-center.at
[5]Software Engineering Group, Leibniz Universität Hannover
Welfengarten 1, D-30167 Hannover, Germany
Kurt.Schneider@inf.uni-hannover.de

Abstract: Requirements engineering has grown into a focus topic for most software-dependent companies. Both outsourcing and in-house development call for effective elicitation of requirements, and for rich communication between customers and software developers. Organizational learning is, therefore, a natural complement when we discuss requirements engineering practice and its improvement. Through organizational learning, processes and tools are systematically improved, reflection and explicit learning becomes part of the company culture. Many companies are still struggling to reach this goal.
The LSO+RE workshop has provided a forum for discussing the intersection of requirements engineering and learning software organizations in depth. This article introduces the topic and the articles from the LSO+RE workshop that have been selected for this special issue of J.UKM.

Keywords: requirements engineering, software process improvement, knowledge management, lessons learned, best practices, learning software organization
Categories: D.2.1, D.2.9, K.6.1, I.2.6

1 Introduction

Software development is an area with a history of cost and time overruns. Many solutions have been proposed to solving problems during the years. Knowledge management has been one area that has been discussed recently [2, 13]. This is important for software companies, since software development is knowledge work,

where employee's knowledge is the critical factor, not other means of production like hardware and development tools. In a competitive environment with constant technological changes, it is a challenge to make room for reflection in the daily work to stimulate learning. Some claim that software organizations suffer from "learning failure" in that they fail to learn from their experience in software development, and have also "learned to fail" [14].

This article introduces a special issue compiled of revised articles from the First International Workshop on Learning Software Organizations and Requirements Engineering (LSO+RE)[1]. LSO+RE was held at Universität Hannover, Germany, on March 27-28, 2006. The workshop fosters interdisciplinary experience sharing on topics such as software process improvement, personal competence development, technology transfer and innovation management, socio-psychological aspects of learning, as well as enabling technology – all related to requirements engineering. We organized that workshop, because requirements engineering conferences rarely focus on learning aspects, and requirements engineering is one of the most vital and learning-oriented processes of software engineering.

In particular in requirements engineering, there is a large potential for improved efficiency, product quality, customer satisfaction and profitability by learning from and using earlier experiences, and by continuously maintaining and refining the accumulated competence of the company. Nevertheless, research documents a low level of systematic use of experiences and knowledge management. An investigation among large European research- and development companies show that only 20 % have reviews of completed projects, and even fewer have good techniques for the reviews [23].

Requirements engineering is particularly important to software engineering, since it is important to understand the needs of the customer who is purchasing a software product. This has been called requirements engineering. Both outsourcing and in-house development call for effective elicitation of requirements, and for rich communication between customer and software developers. Adequate requirements come from appropriate requirements engineering techniques and processes. They are at the basis of successful software projects.

However, requirements engineering is a so-called "wicked problem": by trying to solve it, we create new problems. Documenting requirements well takes time. During that time, requirements changes or re-interpretation of existing requirements may escape our attention. On the other side, poorly documented requirements cause problems in testing and acceptance phases.

Wicked problems are ill-framed and ill-defined; there is no one correct and optimal solution, but an on-going process of negotiations and learning. The problem evolves together with attempted solutions. Organizational learning is, therefore, a natural complement when we discuss requirements engineering. A learning software organization relies heavily on competent and learning individuals – but it will also go beyond the individual learning level: Truly learning organizations are characterized

[1] The LSO+RE workshop is a spin-off of the Learning Software Organizations (LSO) workshop series, which is a forum for software professionals and researchers interested in organizational learning within software development environments. (http://www.iese.fhg.de/Publications/lso/)

by their attempts to foster and support learning on both individual and structure levels ("organizational"). Processes and tools are systematically improved, reflection and explicit learning is part of the company culture. Many companies are still struggling to reach this goal.

A study shows that companies who are better at integrating domain and technical knowledge have increased development effectiveness, and lower cost overruns [21]. Further, in a study of success factors of software process improvement [10], the alignment of software process improvement actions and business strategy and goals was found to be one of the factors with the strongest influence on success. This means that software and business executives should aim more at shared domain knowledge.

The topics under discussion at the LSO+RE workshop were:

- Improving RE practices based on previous experiences,
- Tailoring and adapting RE practices in software organizations,
- Learning environments and material for RE,
- Learning from failure and from success in RE,
- Feedback mechanisms and techniques within RE,
- Collaborative learning of customers and developers and
- Short-term and long-term learning in RE.

As workshop organizers, we hope LSO+RE has helped providing better practices for requirements engineering, increased the usage of well-functioning requirements engineering methods and inspired people to experiment and learn from innovative techniques.

The rest of this article is organized as follows: In section 2, we describe what we mean by a learning software organization, give a brief status of the field, and develop a way to categorize work in this area. Section 3 gives a brief overview of the main topics and challenges in requirements engineering. Section 4 presents the articles selected for this special issue, and places them in the context of organizational learning and requirements engineering. Finally, in Section 5, we summarize the findings from the articles and discussions from the workshop. We give our thoughts on future development in the intersection between learning software organizations and requirements engineering.

2 LSO: The Vision of a Software Organization that Learns

A learning software organization is an organization that develops or maintains software and intentionally acts as a "learning organization". There are many definitions on what is required for a "learning organization" or "organizational learning". In a review article, Dodgson [9] describes learning organizations as organizations that "build, supplement and organize knowledge and routines around their activities and within their cultures, and adapt and develop organizational efficiency by improving the use of the broad skills of their workforces". He further writes that this definition incorporates the following assumptions:

- Learning generally has positive consequences even though learning can be caused by failure.
- Corporate and group culture is influenced by individual learning and can assist the direction of the learning.

- Learning occurs throughout all activities of an organization. Encouraging and coordinating the variety of interactions in learning is a key organizational task.

"Learning organizations" are then organizations that "purposefully construct structures and strategies as to enhance and maximize organizational learning".

Studies on organizational learning in general build on theories from different disciplines, from Argyris and Schön's theory of learning [1], Nonaka and Takeuchi's theories of knowledge creation [15] to Wenger's theories on Communities of Practice [22] and Senge's work on organizational learning [18]. We see knowledge management [5] as a field that covers all the previously mentioned theories.

When we discuss organizational learning in software organizations, this intersects to a high degree with the subfield of software engineering called *software process* improvement. It includes the Total Quality Management (TQM) "philosophy" [7] and it's version related to software: The Quality Improvement Paradigm (QIP). Both fields focus on organizational learning [3]. The concept of the Experience Factory [4] is the first well known systematic approach to organizational learning in the software engineering field. However, the main focus of the literature in software engineering has been on technological issues [8].

So, what makes learning *software* organizations different from other learning organizations? First, software development is a very knowledge-intensive form of work, second software organizations have a higher maturity on information technology usage, so we might expect this type of organizations to make better use of available tools.

Earl [11] has developed a framework to place studies on knowledge management according to different schools, as shown in Table 1. The "technocratic" approach to knowledge management is comprised of works focusing on systems, cartography, and engineering, the "economic" school looks at the commercial value of knowledge, while the "behavioural" school focuses on organizational, spatial and strategic areas.

School / Attribute	**Technocratic**			**Economic**	**Behavioural**		
	Systems	*Cartographic*	*Engineering*	*Commercial*	*Organizational*	*Spatial*	*Strategic*
Focus	Technology	Maps	Processes	Income	Networks	Space	Mindset
Aim	Knowledge bases	Knowledge directories	Knowledge flows	Knowledge assets	Knowledge pooling	Knowledge exhange	Knowledge capabilities
Unit	Domain	Enterprise	Activity	Know-how	Communities	Place	Business
Philo-sophy	Codification	Connectivity	Capability	Commercia-lization	Collaboration	Contact-ivity	Conscious-ness

Table 1: Schools in knowledge management research, developed by Earl.

When a company builds up experience exploitation and organizational learning, a number of expectations are common: Experience seems to be everywhere, so there should be no problem collecting and distributing it to others. However, some of those expectations turn out to be unrealistic. For example, it is unrealistic to assume altruism by those who have knowledge and experience. Why should they share it?

Why should they even invest time and effort to structure or document it? When unrealistic assumptions are made, the success of the initiative is at stake. Schneider [17] reports on an experience exploitation initiative at DaimlerChrysler. Observations include: (1) not only were many software experts reluctant to share experience, they were not enthusiastic about reusing (adopting) it either. (2) The impact of tools was largely overestimated, especially during the early phases of experience exchange. It turned out to be much more crucial to stimulate and activate valuable experiences than to organize or search them in a sophisticated tool. Working on a tool may distract attention from more essential tasks. (3) Soliciting experiences and turning them into best practices was underestimated both in terms of effort and impact. In general, learning software organizations need to take into account many technical details, psychological factors, and organization issues. Reuse of valuable knowledge is possible, but it is not as straight-forward as it may seem.

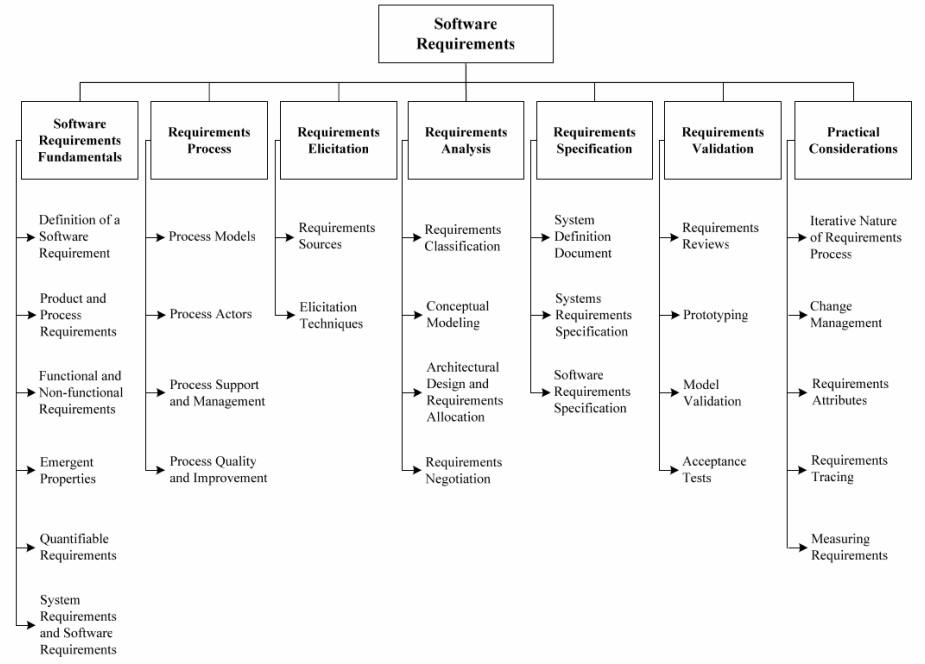

Figure 1: Topics breakdown of the SWEBOK software requirements knowledge area.

3 RE: Learning about Requirements Engineering

Requirements engineering is a field which covers a large variety of works, from technical tools to help analyze and organize requirements, to works on organizational development in order to establish new routines and practices to discuss requirements with a customer.

The IEEE Computer Society's Software Engineering Body of Knowledge (SWEBOK) [19] defines requirements engineering as the area of software

engineering concerned with the acquisition, analysis, specification, validation, and management of software requirements. Figure 1 gives an overview of the topic areas that constitute the field of RE.

Kauppinen et al. [12] have identified the following factors in the requirements engineering literature that influence improvement. Many of these factors are related to learning:

- *User involvement* – involving the stakeholders who will be affected by a changed requirements process.
- *Benefits of the requirements engineering process* – everyone involved should see some benefits.
- *Cultural change* – the development personnel needs to understand importance of customer and user requirements.
- *Continuous requirements engineering improvement* – feedback on the requirements process must be collected and used to take action.
- *Evolutionary requirements engineering process improvement* – use small-scale improvements before new expensive techniques.
- *Pilot projects* – ensure that new processes are feasible through real pilot projects.
- *Training and education* – this is essential for ensuring sustained change.
- *Simplicity of the requirements engineering process* – multiple stakeholders from several functional organizations requires a simple interface.

4 LSO+RE: Contributions to the Intersection

The LSO+RE workshop stimulated the investigation of the manyfold intersections between learning software organization and requirements engineering. In order to structure the discourse, the workshop organizers have proposed the following structure of this intersection area:

- Organizing and managing requirements knowledge
- Organizing and managing knowledge and experience about RE
- Understanding information flows in RE
- Improving RE practices with the aid of knowledge management and communication concepts
  - Learning about RE
  - Applying the learned knowledge

Organizing and managing requirements knowledge is part of the day-to-day project work in RE. How shall we structure system requirements? How shall we document the results of requirements elicitation? How shall we present the knowledge encoded into requirements to the various system and project stakeholders?

Organizing and managing knowledge and experience about RE is a task relevant to RE education, job profiles, task assignment and process definition, as well as RE process improvement. It covers the application of LSO to the process and how-to aspects of RE. It focuses on the explicitly documented knowledge about RE, but also considers the tacit parts of the knowledge.

Understanding information flows in RE is concerned with making explicit the many tacit and implicit knowledge aspects of the RE practice. It provides the basis for

making this knowledge explicit and manageable by a larger set of knowledge management instruments.

Improving RE practices with the aid of knowledge management and communication concepts is concerned with the question how we can gain and utilize knowledge about RE. It consists of the two complimentary activitis of learing about RE, i.e., gaining knowledge, and applying the learned knowledge, mostly in the realms of education and process improvement.

This structuring of the intersection between LSO and RE provides a framework for categorizing the articles selected for this special issue. Table 2 relates each article to the respective categories. The category *organizing and managing requirements knowledge* is not addressed by any of these articles. However, there were three additional workshop contributions that placed special emphasis on this category: Decker et al. [6], Nicolás et al. [16], and Tavakoli and Porta [20].

Organizing and managing knowledge and experience about RE	Understanding information flows in RE	Improving RE practices with the aid of KM and communication concepts
Hagge and Lappe Herrmann Houdek	Allmann et al. Knauss et al.	Fricker et al. Smolander Šmite Werner

Table 2: Relations of the articles in this special issue to LSO+RE categories.

According to the other LSO frameworks outlined above, the articles can be characterized as follows. If we place the articles according to the knowledge management schools, we see that all articles fits into an "engineering" school, which is not surprising given the overall topic of software engineering. The articles tend to focus on the requirements engineering process, and either describe it in different contexts, such as a global context (Šmite) and viewing it like a process of interpretations by different parties (Ovaska et al.). Other articles present improvement projects aimed at formalizing requirements engineering: Werner, Allmann et al. and Fricker et al. The article by Knauss et al. describes an effort to balance tacit and explicit knowledge in the requirements process, while Hagge and Lappe discuss codified patterns as a means to learn from previous efforts. Finally, Houdek discusses organizational learning techniques in relation to suitability for requirements engineering. One of the approaches Houdek discusses is a knowledge base, which is also a theme in the article by Allmann et al. This is a theme that would fit in the "systems" school in Earl's knowledge management framework.

If we place the articles according to success factors in requirements engineering, we see that several of the articles address training, which is one of the success factors in our framework: Šmite discusses the role of training on requirements engineering in global projects, while Houdek discusses both coaching and internal training as possible candidates of learning strategies on requirements engineering. Werner emphasize the importance of training the users, which correspond to user involvement as well as training and education. Ovaska et al. found four categories of expectations and attitudes to requirements: business value, system development strategy, system

development capability and system development resource allocation. Knauss et al. discusses the balance of documentation, which is related to the simplicity of the requirements engineering process.

5 LSO+RE 2006: Status and Outlook

The articles presented at LSO+RE and the discussions among participants have shown that the field of RE calls for a broad spectrum of support from organizational learning. Much of this need persists. The concept of learning organization should be disseminated further and adapted more closely to requirements engineering.

Workshop contributions from industrial practice have shown that organizational learning techniques can be beneficial to day-to-day software development practice. First of all, the basic principles of organizational learning and an appropriate organizational culture should be introduced and implemented within a software organization. Then, appropriate techniques and processes that facilitate learning need to be adapted to the specific characteristics of the software organization.

Some controversial discussion span off around the question whether knowledge management should focus on explicit knowledge representation such as process definition and best practice repositories, or whether organizational learning should predominantly proceed through informal stakeholder interaction. The discussion participants presented evidence that the importance of explicit knowledge representation tends to be overestimated. However, appropriate and sound documentation are important ingredients of organizational learning, in particular in order to bridge access to information over larger time spans or large geographical distance. For ensuring actual application of knowledge and experience, personal contact and assistance appeared to be an important prerequisite.

The articles presented in this special issue focus primarily on the "engineering" school of knowledge management, and partially on the "systems" school. They focus on a broad set of topics within requirements engineering, including user involvement, training and education. They argue for a simple but carefully designed requirements engineering process that includes the right principles and techniques from learning software organizations.

In the future, studies from other schools of knowledge management may enrich discussion. A map of knowledge management within requirements engineering could be an important topic, as raised by Houdek. Also, organizational and spatial approaches are interesting paths to follow, particularly in environments primarily relying on the transfer of tacit knowledge as in companies focusing on agile development.

Acknowledgements

As workshop organizers and editors of this special issue of J.UKM, we want to thank all workshop participants and authors of workshop articles for their highly interesting and valuable contributions. We enjoyed the open atmosphere during LSO+RE in Hannover, which made it a rewarding event.

The Know-Center is funded by the Austrian Competence Center program Kplus under the auspices of the Austrian Ministry of Transportation, Innovation and Technology (www.ffg.at/index.php?cid=95) and by the State of Styria.

References

[1] Chris Argyris and Donald A. Schön, *Organizational Learning II: Theory, Method and Practise*: Addison Wesley, 1996,

[2] A. Aurum, R. Jeffery, C. Wohlin, and M. Handzic, *Managing Software Engineering Knowledge*. Berlin: Springer Verlag, 2003,

[3] Victor R. Basili, "Quantitative Evaluation of Software Engineering Methodology", Proceedings of the First Pan Pacific Computer Conference, Melbourne, Australia, 1985.

[4] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach, "The Experience Factory", in *Encyclopedia of Software Engineering*, vol. 1, J. J. Marciniak, Ed.: John Wiley, 1994, pp. 469-476.

[5] Thomas H Davenport and Laurence Prusak, *Working Knowledge: How Organizations Manage What They Know*: Harvard Business School Press, 1998, ISBN 0-87584-655-6.

[6] Björn Decker, Eric Ras, Jörg Rech, Bertin Klein, Christian Hoecht, "Using Wikis to Manage Use Cases: Experience and Outlook": *Proceedings of the International Workshop on Learning Software Organizations and Requirements Engineering*, University of Hannover, March 27 and 28, 2006.

[7] Edwards W. Deming, *Out of the Crisis*. Cambridge, Massachusetts: The MIT Press (first published in 1982 by MIT Center for Advanced Educational Services), 2000, ISBN 0-262-54115-7.

[8] Torgeir Dingsøyr and Reidar Conradi, "A Survey of Case Studies of the Use of Knowledge Management in Software Engineering", *International Journal of Software Engineering and Knowledge Engineering*, no. 4, vol. 12, pp. 391 - 414, 2002.

[9] Mark Dodgson, "Organizational Learning: A Review of Some Literatures", *Organizational Studies*, no. 3, vol. 14, pp. 375 - 394, 1993.

[10] Tore Dybå, "An Empirical Investigation of the Key Factors for Success in Software Process Improvement", *IEEE Transactions on Software Engineering*, no. 5, vol. 31, pp. 410 - 424, 2005.

[11] Michael Earl, "Knowledge Mangement Strategies: Towards a Taxonomy", *Journal of Management Information Systems*, no. 1, vol. 18, 2001.

[12] Marjo Kauppinen, Matti Vartiainen, Jyrki Kontio, Sari Kujala, and Reijo Sulonen, "Implementing requirements engineering processes through organizations: success factors and challenges", *Information and Software Technology*, vol. 46, pp. 937 - 953, 2004.

[13] Mikael Lindvall and Ioana Rus, "Knowledge Management in Software Engineering", *IEEE Software*, no. 3, vol. 19, pp. 26 - 38, 2002.

[14] K. Lyytinen and D. Robey, "Learning Failure in information systems development", *Information Systems Journal*, vol. 9, pp. 85 -101, 1999.

[15] Ikujiro Nonaka and Hirotaka Takeuchi, *The Knowledge-Creating Company*: Oxford University Press, 1995, ISBN 0-18.509269-4.

[16] Joaquín Nicolás, Joaquín Lasheras, Ambrosio Toval, Francisco J. Ortiz, Bárbara Álvarez, " An Experience on Modelling Teleoperated Systems for Ship Coating Removal through Features and Generic Use Cases ": *Proceedings of the International Workshop on Learning Software Organizations and Requirements Engineering*, University of Hannover, March 27 and 28, 2006.

[17] Kurt Schneider, "What to Expect from Software Experience Exploitation," Proc. of the IKnow Conference, Graz, Austria: *Journal of Universal Computer Science J.UCS* 8(6), www.jucs.org, , 44-54.

[18] Peter M. Senge, *The Fifth Discipline: The Art & Practise of The Learning Organisation*: Century Business, 1990, ISBN 0-7126-56871.

[19] Pierre Bouque and Robert Dupuis (Eds.), *Guide to the Software Engineering Body of Knowledge, 2004 Version (SWEBOK)*: IEEE Computer Society, 2000, http://www.swebok.org.

[20] Ramin Tavakoli Kolagari and Nicolas Fernando Porta, "Project-oriented Reuse Approaches: Copy-and-Paste or Software Product Line Engineering?": *Proceedings of the International Workshop on Learning Software Organizations and Requirements Engineering*, University of Hannover, March 27 and 28, 2006.

[21] Amrit Tiwana, "An empirical study of the effect of knowledge integration on software development projects," *Information and Software Technology*, vol. 46, pp. 899 - 906, 2004.

[22] Etienne Wenger, *Communities of practise : learning, meaning and identity*. Cambridge, UK: Cambridge University Press, 1998, ISBN 0-521-43017-8.

[23] Maximilian Zedtwitz, "Organizational learning through post-project reviews in R&D," *R&D Management*, no. 3, vol. 32, pp. 255 - 268, 2002.