Grigori Melnik
Harald Holz (Eds.)

LNCS 3096

Advances in Learning Software Organizations

6th International Workshop, LSO 2004
Banff, Canada, June 2004
Proceedings

Springer

Grigori Melnik   Harald Holz (Eds.)

Advances in Learning Software Organizations

6th International Workshop, LSO 2004
Banff, Canada, June 20-21, 2004
Proceedings

Springer

Volume Editors

Grigori Melnik
University of Calgary, Department of Computer Science
2500 University Dr. N.W., Calgary, Alberta T2N 1N4, Canada
E-mail: melnik@cpsc.ucalgary.ca

Harald Holz
German Research Center for Artificial Intelligence (DFKI) GmbH
Knowledge Management Department
Erwin-Schroedinger-Str. 57, 67663 Kaiserslautern, Germany
E-mail: harald.holz@dfki.de

Library of Congress Control Number: 2004106916

CR Subject Classification (1998): D.2, K.6, H.5.2-3, I.2.4, K.3, K.4.3

ISSN 0302-9743
ISBN 3-540-22192-1 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag is a part of Springer Science+Business Media

springeronline.com

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11013808 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3096

Preface

Software-intensive organizations cannot help but learn. A software organization that does not learn will not exist for long, because the software market is continuously on the move, because of new customer demands and needs, and because of new competitor products and services. Software organizations must adapt quickly to this ever-changing environment, and the capability to adapt is one of the most important aspects of learning. Smart organizations will attempt to predict future software demands, and develop a corresponding knowledge road map that identifies the capabilities needed over time in order to meet these demands.

Organizational learning typically occurs when experienced organization members share their knowledge with colleagues, such that the organization as a whole can profit from the intellectual capital of its members. While knowledge is typically shared in an ad hoc fashion by means of direct, face-to-face communication, a learning software organization will want to ensure that this knowledge sharing occurs in a systematic way, enabling it whenever and wherever it is needed.

Since 1999, the annual International Workshop on Learning Software Organizations (LSO) has provided a communication forum that brings together academia and industry to discuss the advancements in and to address the questions of continuous learning in software-intensive organizations. Building upon existing work on knowledge management and organizational learning, the workshop series promotes interdisciplinary approaches from computer science and information systems, business, management and organization science as well as cognitive science.

The LSO workshop series differs from other conferences in that it puts a particular focus on organizational (rather than mere technological) questions such as: How to introduce knowledge management approaches in a software organization in minimally intrusive ways? What IT support is successful in fostering a culture of knowledge sharing? What roles are required to effectively disseminate available knowledge? How can information overload be avoided, while ensuring that relevant information is not overlooked? What knowledge management approaches scale up to the needs of large, multisite organizations? Approaches that address these issues often have to integrate a number of diverse techniques, methods, and tools.

This volume contains 13 full papers and 3 short papers selected by the program committee for presentation at the 6th International Workshop on Learning Software Organizations (LSO 2004). The workshop was held in conjunction with the 16th International Conference on Software Engineering and Knowledge Engineering (SEKE 2004) which allowed for further fruitful, inter-disciplinary discussions between both groups of participants. In addition to the oral presentation of the papers contained in this volume, the LSO 2004 program included a keynote talk by Philippe Kruchten.

As the workshop chairs, we would like to thank the authors and presenters for their willingness to share their expertise and report on their latest results. Moreover, our thanks go to the members of the program committee who did a great job reviewing the submitted papers. Last but not least, we would especially like to thank our keynote speaker Philippe Kruchten for sharing his insights with us.

We hope that this volume conveys to its readers at least some of the vital and interesting discussions that were stimulated by the presentation given at the LSO 2004 workshop.

June 2004

Grigori Melnik
Harald Holz

Workshop Organization

Workshop Chairs

Grigori Melnik, University of Calgary, Calgary (Canada)
Harald Holz, DFKI GmbH, Kaiserslautern (Germany)
Scott Henninger, University of Nebraska-Lincoln, Lincoln (USA)

Program Committee

Klaus-Dieter Althoff, Fraunhofer IESE (Germany)
Ralph Bergmann, University of Hildesheim (Germany)
Andreas Birk, sd&m AG (Germany)
Raimund L. Feldmann, Fraunhofer Center Maryland (USA)
Christiane Gresse von Wangenheim, Universidade do Vale do Itajaí (Brazil)
John C. Grundy, University of Auckland (New Zealand)
Frank Houdek, DaimlerChrysler AG (Germany)
Franz Lehner, University of Regensburg (Germany)
Mikael Lindvall, Fraunhofer Center Maryland (USA)
Makoto Matsushita, Osaka University (Japan)
Frank Maurer, University of Calgary (Canada)
Günther Ruhe, University of Calgary (Canada)
Ioana Rus, Fraunhofer Center Maryland (USA)
Martin Schaaf, University of Hildesheim (Germany)
Markus Strohmaier, The Know-Center, Graz (Austria)
Carsten Tautz, empolis GmbH (Germany)
Rosina Weber, Drexel University (USA)
Yunwen Ye, University of Colorado at Boulder (USA)

Additional Reviewers

Michael M. Richter, University of Kaiserslautern (Germany)
Torsten Willrich, Fraunhofer IESE (Germany)

Steering Committee

Klaus-Dieter Althoff, Fraunhofer IESE (Germany)
Raimund L. Feldmann, Fraunhofer Center Maryland (USA)
Scott Henninger, University of Nebraska-Lincoln (USA)
Frank Maurer, University of Calgary (Canada)
Wolfgang Müller, University of Applied Science, Ludwigshafen (Germany)

Sponsoring Institutions

The organizers of LSO 2004 would like to thank the following institutions for their generous support:

- informatics Circle of Research Excellence (iCORE), Canada
- University of Calgary, Canada
- German Research Center for Artificial Intelligence (DFKI) GmbH, Germany

Furthermore, the organizers thank for administrative support and logistics:

- Camille Sinanan, University of Calgary (Canada)
- The Banff Centre (Canada)

LSO Workshop Series

Materials of past and upcoming workshops in the Learning Software Organizations series are available from:

http://www.iese.fhg.de/Publications/lso

Lecture Notes in Computer Science

For information about Vols. 1–2988

please contact your bookseller or Springer-Verlag

Vol. 3096: G. Melnik, H. Holz (Eds.), Advances in Learning Software Organizations. X, 173 pages. 2004.

Vol. 3093: S.K. Katsikas, S. Gritzalis, J. Lopez (Eds.), Public Key Infrastructure. XIII, 380 pages. 2004.

Vol. 3092: J. Eckstein, H. Baumeister (Eds.), Extreme Programming and Agile Processes in Software Engineering. XVI, 358 pages. 2004.

Vol. 3091: V. van Oostrom (Ed.), Rewriting Techniques and Applications. X, 313 pages. 2004.

Vol. 3089: M. Jakobsson, M. Yung, J. Zhou (Eds.), Applied Cryptography and Network Security. XIV, 510 pages. 2004.

Vol. 3085: S. Berardi, M. Coppo, F. Damiani (Eds.), Types for Proofs and Programs. X, 409 pages. 2004.

Vol. 3084: A. Persson, J. Stirna (Eds.), Advanced Information Systems Engineering. XIV, 596 pages. 2004.

Vol. 3083: W. Emmerich, A.L. Wolf (Eds.), Component Deployment. X, 249 pages. 2004.

Vol. 3078: S. Cotin, D.N. Metaxas (Eds.), Medical Simulation. XVI, 296 pages. 2004.

Vol. 3077: F. Roli, J. Kittler, T. Windeatt (Eds.), Multiple Classifier Systems. XII, 386 pages. 2004.

Vol. 3076: D. Buell (Ed.), Algorithmic Number Theory. XI, 451 pages. 2004.

Vol. 3074: B. Kuijpers, P. Revesz (Eds.), Constraint Databases and Applications. XII, 181 pages. 2004.

Vol. 3073: H. Chen, R. Moore, D.D. Zeng, J. Leavitt (Eds.), Intelligence and Security Informatics. XV, 536 pages. 2004.

Vol. 3070: L. Rutkowski, J. Siekmann, R. Tadeusiewicz, L.A. Zadeh (Eds.), Artificial Intelligence and Soft Computing - ICAISC 2004. XXV, 1208 pages. 2004. (Subseries LNAI).

Vol. 3068: E. André, L. Dybkj\ae r, W. Minker, P. Heisterkamp (Eds.), Affective Dialogue Systems. XII, 324 pages. 2004. (Subseries LNAI).

Vol. 3066: S. Tsumoto, R. S lowiński, J. Komorowski, J.W. Grzymala-Busse (Eds.), Rough Sets and Current Trends in Computing. XX, 853 pages. 2004. (Subseries LNAI).

Vol. 3065: A. Lomuscio, D. Nute (Eds.), Deontic Logic in Computer Science. X, 275 pages. 2004. (Subseries LNAI).

Vol. 3064: D. Bienstock, G. Nemhauser (Eds.), Integer Programming and Combinatorial Optimization. XI, 445 pages. 2004.

Vol. 3063: A. Llamosí, A. Strohmeier (Eds.), Reliable Software Technologies - Ada-Europe 2004. XIII, 333 pages. 2004.

Vol. 3062: J.L. Pfaltz, M. Nagl, B. Böhlen (Eds.), Applications of Graph Transformations with Industrial Relevance. XV, 500 pages. 2004.

Vol. 3060: A.Y. Tawfik, S.D. Goodwin (Eds.), Advances in Artificial Intelligence. XIII, 582 pages. 2004. (Subseries LNAI).

Vol. 3059: C.C. Ribeiro, S.L. Martins (Eds.), Experimental and Efficient Algorithms. X, 586 pages. 2004.

Vol. 3058: N. Sebe, M.S. Lew, T.S. Huang (Eds.), Computer Vision in Human-Computer Interaction. X, 233 pages. 2004.

Vol. 3056: H. Dai, R. Srikant, C. Zhang (Eds.), Advances in Knowledge Discovery and Data Mining. XIX, 713 pages. 2004. (Subseries LNAI).

Vol. 3054: I. Crnkovic, J.A. Stafford, H.W. Schmidt, K. Wallnau (Eds.), Component-Based Software Engineering. XI, 311 pages. 2004.

Vol. 3053: C. Bussler, J. Davies, D. Fensel, R. Studer (Eds.), The Semantic Web: Research and Applications. XIII, 490 pages. 2004.

Vol. 3052: W. Zimmermann, B. Thalheim (Eds.), Abstract State Machines 2004. Advances in Theory and Practice. XII, 235 pages. 2004.

Vol. 3051: R. Berghammer, B. Möller, G. Struth (Eds.), Relational and Kleene-Algebraic Methods in Computer Science. X, 279 pages. 2004.

Vol. 3050: J. Domingo-Ferrer, V. Torra (Eds.), Privacy in Statistical Databases. IX, 367 pages. 2004.

Vol. 3047: F. Oquendo, B. Warboys, R. Morrison (Eds.), Software Architecture. X, 279 pages. 2004.

Vol. 3046: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1016 pages. 2004.

Vol. 3045: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1040 pages. 2004.

Vol. 3044: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1140 pages. 2004.

Vol. 3043: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1180 pages. 2004.

Vol. 3042: N. Mitrou, K. Kontovasilis, G.N. Rouskas, I. Iliadis, L. Merakos (Eds.), NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications. XXXIII, 1519 pages. 2004.

Vol. 3039: M. Bubak, G.D.v. Albada, P.M. Sloot, J.J. Dongarra (Eds.), Computational Science - ICCS 2004. LXVI, 1271 pages. 2004.

Vol. 3038: M. Bubak, G.D.v. Albada, P.M. Sloot, J.J. Dongarra (Eds.), Computational Science - ICCS 2004. LXVI, 1311 pages. 2004.

Vol. 3037: M. Bubak, G.D.v. Albada, P.M. Sloot, J.J. Dongarra (Eds.), Computational Science - ICCS 2004. LXVI, 745 pages. 2004.

Vol. 3036: M. Bubak, G.D.v. Albada, P.M. Sloot, J.J. Dongarra (Eds.), Computational Science - ICCS 2004. LXVI, 713 pages. 2004.

Vol. 3035: M.A. Wimmer (Ed.), Knowledge Management in Electronic Government. XII, 326 pages. 2004. (Subseries LNAI).

Vol. 3034: J. Favela, E. Menasalvas, E. Chávez (Eds.), Advances in Web Intelligence. XIII, 227 pages. 2004. (Subseries LNAI).

Vol. 3033: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVIII, 1076 pages. 2004.

Vol. 3032: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVII, 1112 pages. 2004.

Vol. 3031: A. Butz, A. Krüger, P. Olivier (Eds.), Smart Graphics. X, 165 pages. 2004.

Vol. 3030: P. Giorgini, B. Henderson-Sellers, M. Winikoff (Eds.), Agent-Oriented Information Systems. XIV, 207 pages. 2004. (Subseries LNAI).

Vol. 3029: B. Orchard, C. Yang, M. Ali (Eds.), Innovations in Applied Artificial Intelligence. XXI, 1272 pages. 2004. (Subseries LNAI).

Vol. 3028: D. Neuenschwander, Probabilistic and Statistical Methods in Cryptology. X, 158 pages. 2004.

Vol. 3027: C. Cachin, J. Camenisch (Eds.), Advances in Cryptology - EUROCRYPT 2004. XI, 628 pages. 2004.

Vol. 3026: C. Ramamoorthy, R. Lee, K.W. Lee (Eds.), Software Engineering Research and Applications. XV, 377 pages. 2004.

Vol. 3025: G.A. Vouros, T. Panayiotopoulos (Eds.), Methods and Applications of Artificial Intelligence. XV, 546 pages. 2004. (Subseries LNAI).

Vol. 3024: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3023: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 611 pages. 2004.

Vol. 3022: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3021: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 633 pages. 2004.

Vol. 3019: R. Wyrzykowski, J.J. Dongarra, M. Paprzycki, J. Wasniewski (Eds.), Parallel Processing and Applied Mathematics. XIX, 1174 pages. 2004.

Vol. 3018: M. Bruynooghe (Ed.), Logic Based Program Synthesis and Transformation. X, 233 pages. 2004.

Vol. 3016: C. Lengauer, D. Batory, C. Consel, M. Odersky (Eds.), Domain-Specific Program Generation. XII, 325 pages. 2004.

Vol. 3015: C. Barakat, I. Pratt (Eds.), Passive and Active Network Measurement. XI, 300 pages. 2004.

Vol. 3014: F. van der Linden (Ed.), Software Product-Family Engineering. IX, 486 pages. 2004.

Vol. 3012: K. Kurumatani, S.-H. Chen, A. Ohuchi (Eds.), Multi-Agnets for Mass User Support. X, 217 pages. 2004. (Subseries LNAI).

Vol. 3011: J.-C. Régin, M. Rueher (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 415 pages. 2004.

Vol. 3010: K.R. Apt, F. Fages, F. Rossi, P. Szeredi, J. Váncza (Eds.), Recent Advances in Constraints. VIII, 285 pages. 2004. (Subseries LNAI).

Vol. 3009: F. Bomarius, H. Iida (Eds.), Product Focused Software Process Improvement. XIV, 584 pages. 2004.

Vol. 3008: S. Heuel, Uncertain Projective Geometry. XVII, 205 pages. 2004.

Vol. 3007: J.X. Yu, X. Lin, H. Lu, Y. Zhang (Eds.), Advanced Web Technologies and Applications. XXII, 936 pages. 2004.

Vol. 3006: M. Matsui, R. Zuccherato (Eds.), Selected Areas in Cryptography. XI, 361 pages. 2004.

Vol. 3005: G.R. Raidl, S. Cagnoni, J. Branke, D.W. Corne, R. Drechsler, Y. Jin, C.G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G.D. Smith, G. Squillero (Eds.), Applications of Evolutionary Computing. XVII, 562 pages. 2004.

Vol. 3004: J. Gottlieb, G.R. Raidl (Eds.), Evolutionary Computation in Combinatorial Optimization. X, 241 pages. 2004.

Vol. 3003: M. Keijzer, U.-M. O'Reilly, S.M. Lucas, E. Costa, T. Soule (Eds.), Genetic Programming. XI, 410 pages. 2004.

Vol. 3002: D.L. Hicks (Ed.), Metainformatics. X, 213 pages. 2004.

Vol. 3001: A. Ferscha, F. Mattern (Eds.), Pervasive Computing. XVII, 358 pages. 2004.

Vol. 2999: E.A. Boiten, J. Derrick, G. Smith (Eds.), Integrated Formal Methods. XI, 541 pages. 2004.

Vol. 2998: Y. Kameyama, P.J. Stuckey (Eds.), Functional and Logic Programming. X, 307 pages. 2004.

Vol. 2997: S. McDonald, J. Tait (Eds.), Advances in Information Retrieval. XIII, 427 pages. 2004.

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2995: C. Jensen, S. Poslad, T. Dimitrakos (Eds.), Trust Management. XIII, 377 pages. 2004.

Vol. 2994: E. Rahm (Ed.), Data Integration in the Life Sciences. X, 221 pages. 2004. (Subseries LNBI).

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2990: J. Leite, A. Omicini, L. Sterling, P. Torroni (Eds.), Declarative Agent Languages and Technologies. XII, 281 pages. 2004. (Subseries LNAI).

Vol. 2989: S. Graf, L. Mounier (Eds.), Model Checking Software. X, 309 pages. 2004.

Table of Contents

Introduction

Experience-Based Information Systems

Software Maintenance

Communities of Practice

Planning LSOs

Case Studies and Experience Reports

Research on Learning Software Organizations – Past, Present, and Future

Harald Holz[1] and Grigori Melnik[2]

[1] DFKI GmbH, Knowledge Management Department,
PO Box 2080, 67608 Kaiserslautern, Germany
Harald.Holz@dfki.de
[2] Department of Computer Science, University of Calgary,
Calgary, Canada
melnik@cpsc.ucalgary.ca

Knowledge itself is power, not mere argument or ornament.
Francis Bacon (Meditations Sacrae, 1597)

1 Introduction

In order for a software organization to stay competitive, its software development needs to be part of organizational change. The organization's ability to change and to adapt quickly to environmental changes provides a foundation for growth and power [7]. For such changes to happen, the learning capabilities of the organization have to be enhanced, being an essential part of producing more effective and efficient work practices. Moreover, continuous learning is essential for surviving – let alone prospering – in dynamic and competitive environments [15]. The Learning Software Organization (LSO) workshop series has been promoting this vision since 1999, addressing the questions of organizational learning from the software development point of view.

Though the workshop series is relatively young, the ideas it is based on have been circulating for decades. As early as in 1971, Weinberg recognized software development as learning: "writing a program is a process of learning – both for the programmer and the person who commissions the program" [23]. This was superseded by the engineering approach, when software development began to be considered as "software engineering", omitting for a long period the humanistic people-centric aspect of it. The history of LSO workshops reflects this development to a certain degree. In 1999, LSO started with the premise that "with continuous technological change, globalization, business reorganizations, e-migration, etc. there is a continuous shortage of the right knowledge at the right place at the right time. To overcome this shortage is the challenge for the Learning Organization." [20]. In other words, the main challenge considered six years ago was the *availability* of knowledge. As a result, many solutions were built to address this. The proliferation of knowledge bases is a clear indication of this. The knowledge is primarily considered to be an object, and, thus, it can be codified, stored, retrieved and distributed. Unfortunately, many such solutions suffered from the "build it and they'll come" syndrome which resulted in a lack of user involvement and enthusiasm. Researchers and practitioners in LSO began to realize that knowledge externalization and storage are not automatically equal to knowledge re-use, that building an experience factory for the sake of experience fac-

G. Melnik and H. Holz (Eds.): LSO 2004, LNCS 3096, pp. 1–6, 2004.

tory will not pay off the investment. Thus, new efforts to enhance the utilization of knowledge/experience bases/repositories along with improving the software development process commenced. Years 2000 and 2001 were greatly influenced by the Software Process Improvement (SPI) initiatives. The challenges addressed went beyond the availability of knowledge – but further into its *understandability*, *re-use*, *relevancy* and *applicability*. The LSO 2001 main theme was the enablement of the members of the learning software organizations "to effectively quarrel situational requirements, taking past experience into account. Besides improving internal communication (group learning), this also includes documenting relevant knowledge and storing it (for reuse) in an organizational, corporate memory" [1].

Nowadays, it is commonly recognized that promotion of a learning culture and fostering of the exchange of experiences are imperative. Increasingly, experts agree that approaches to achieve this must be based on interdisciplinary research, taking into account results from economical, organizational, cultural, psychological and technological areas.

The year 2002 was marked by the symbiosis of organizational learning and *agility*, the problems facing both the LSO community and the agile methods community seeming to be complementary [10]. Specifically, for the LSO community, the issue is how to quickly adapt to new technologies and market pressures, while for the agile community the issues are how not to lose institutional knowledge and how to enable inter-team learning. The participants commonly recognized the need for a balance between knowledge capture/dissemination and flexibility that enhances the ability of an organization to quickly adapt. In the meantime, the debate about the epistemology of knowledge – whether knowledge is an object or a relation (a context-bound one) – continues.

In 2003, the workshop progressed into the aspects of the evolution of learning organizations and the resulting evolution of repositories they use. Essentially, the workshop focused on the issues of *maintainability* and *scalability* of externalized knowledge [17].

This year (2004), we continue the advancement of the concepts, approaches and techniques to help learning software organizations succeed. The papers included in this volume clearly build upon the results of the previous five workshops. Though a good portion of research today is still dedicated to the development of knowledge management methods and tools, there is an increasing trend towards knowledge management approaches that are lightweight, i.e., do not introduce a considerable additional burden on developers and end users, while at the same time ensuring that the hoped for experience factories do not become "experience cemeteries" which no employee uses. Consequently, the focus is on practical knowledge management initiatives that:

- allow for an incremental adoption without a large up-front investment;
- are flexible enough to allow quick and easy improvements;
- encompass not only the structure, the strategies and the systems of the learning organization itself, but also of those who develop, follow and utilize these structure, strategies and systems.

The following section briefly summarizes current work reflecting the state-of-art and state-of-practice in learning software organizations as presented at the Sixth International Workshop on Learning Software Organizations (LSO 2004) in Banff, Canada.

2 Current Topics in LSO Research and Practice

The 13 full papers and 3 short papers in this volume are drawn from an international base of authors (53), including Belgium, Brazil, Canada, Germany, Mexico, New Zealand, Norway, Portugal, Spain, Switzerland, and United Arab Emirates. A consistent message across all these diverse contributions is that, in order to be effective and agile, we should consider organizational learning as a holistic process, taking into account the particularities of the organization under consideration. Most concepts, approaches and tools will be applicable only in a certain context. The contributions are organized into the following chapters:

- Experience-Based Information Systems
- Software Maintenance
- Communities of Practice
- Planning LSOs
- Case Studies and Experience Reports

Experience-Based Information Systems

Software development processes consist of various knowledge-intensive tasks during which software engineers need to make informed decisions. The contributions in this chapter describe information systems that support users in their decision-making in diverse tasks such as risk management and COTS selection.

Falbo et al. present an ontology-based to support organizational learning in risk management [5]. Their tool GeRis supports novice project managers in the identification, evaluation, ranking, and contingency planning of risks for a current project by providing the manager with corresponding experience from similar, stored projects.

Santos et al. present an enterprise ontology that provides the basis for various tools as part of an enterprise-oriented software development environment (EOSDE) [21]. They illustrate their approach by describing two tools that make use of this ontology: Sapiens, a corporate 'yellow pages' tool, and RHPlan, a resource allocation planning tool. Their EOSDE is already being used in 18 small and medium-size software companies.

In [9], Gomes et al. describe their tool REBUILDER, a CBR system that supports designers by retrieving former UML designs similar to the current design diagram, and by automatically augmenting the current diagram by missing elements from former designs. Moreover, the system provides functionality to evaluate the resulting diagrams based on various object-oriented metrics.

Mohamed et al. propose a conceptual model to support decision making during COTS selection processes [13]. They outline how this model can be implemented as an agent-based decision-support system that addresses important issues such as changing stakeholder preferences and evaluation process simulation to try out different scenarios.

Ras and Weibelzahl argue that experience packages retrieved from repositories are often inadequate for learning and competence development, e.g., because users might not have sufficient knowledge to understand the package content, or because users might be unsure of the risks involved on applying the packaged experience [16]. Their approach addresses these issues by automatically enriching experience packages with additional learning elements based on didactical considerations.

Software Maintenance

Several studies indicate that the processes needed to correct errors in a software system, or to adapt a system to the ever-changing environment incurs most of the overall expenses during the life-cycle of a software product.

In [22], de Sousa et al. propose to use postmortem analysis (PMA) to help manage the knowledge gained during maintenance projects, both knowledge on the maintenance process itself and on the system maintened. Based on a standardized maintenance process, they detail when to conduct PMA during process execution, what knowledge to look for, and how to perform PMA during maintenance.

Rodríguez et al. outline the architecture of a multi-agent system designed to manage knowledge generated during the software maintenance process [18]. Their web-based system aims at proactively providing maintenance engineers with knowledge sources that could help them in carrying out their current tasks.

In [19], Roth-Berghofer reports on experience gained from setting-up and running an internal CAD/CAM help desk support system for IT-related problems at a large company. He discusses lessons learned from this project, where a systematic maintenance process needed to be established, e.g. in order to enhance the domain model appropriately whenever necessary because of environmental changes.

Communities of Practice

Communities of Practice (CoP) are informal groups of organization members that share common interest, practices, and subjects. Approaches that integrate CoPs into daily work processes by lightweight IT support as well as the advantages of informal knowledge exchange are discussed in this chapter.

Chau and Maurer present the lightweight, Wiki-based knowledge management tools MASE and EB [2]. These tools support agile teams by providing them with a process support systems that enables users to share their experience by a collaborative creation and task-specific retrieval of WIKI pages containing information related to the task type. Moreover, first result from a study on inter-team learning using MASE and EB are reported.

In [14], Montoni et al. present a knowledge management approach for acquiring and preserving knowledge related to specific software processes. Their tool ACKNOWLEDGE supports the capture of different knowledge items such as lessons-learned or ideas, as well as their subsequent evaluation by an evaluation committee, and the packaging by knowledge managers. Knowledge items can be retrieved from a community of practice repository via a web-based system with regard to a given process type, user-specified keywords and knowledge types.

In [12], Melnik and Richter analyze the role of imprecise statements in conversations among software developers. They argue that impreciseness can be very useful in interaction, and describe how finding an optimal level of impreciseness can be interpreted as a learning problem for software organizations.

Planning LSOs

An important characteristic of learning software organizations is that learning processes are systematic – learning should not occur in an ad-hoc, chaotic fashion, but as part of the organization's overall strategy, where continuous learning is identified as an explicit goal and methods are deployed to achieve it.

In order to be agile, integrated and aligned, an organization must be architected accordingly. Therefore, Goethals et al. present their framework FADE for managing the concurrent development of the business and the ICT side of an enterprise [8]. FADE identifies several enterprise life-cycle phases as well as their links to the strategic, tactical, and operational level.

Case Studies and Experience Reports

The contributions in this chapter report on experiences and case studies conducted in an industry context. The discuss successes achieved as well as mistakes made, and outline lessons learned.

In [3], Doran reports on his experience with the implementation of knowledge management techniques in an agile software development department of a start-up company. He outlines the difficulties encountered ant approaches chosen for handling knowledge related to process, problem domain and technology, and discusses the tools introduced into the company to support these approaches.

Based on their experience with an industry partner, Draheim and Weber outline general conditions for an approach to collaborative learning of software organizations and academia [4]. They propose a co-knowledge acquisition and sharing process that is lightweight, peer-to-peer, and demand-driven.

In [6], Folkestad et al. report on a case study on the effect of introducing the Unified Process and object-oriented technologies into a company. The authors demonstrate the application of activity theory in a qualitative approach, and identify the iterative development introduced by the Unified Process to have a large effect on organizational and individual learning, flanked by new roles and more formal communication patterns.

John and Melster report on their experience from building and using a knowledge model for a knowledge network for know-how transfer in the area of software engineering, using a classical approach to model building [11]. Based on this experience, they outline a personal and peer-to-peer knowledge management approach that better takes into account the flexible and social structures of knowledge expert communities.

3 Conclusion

The diversity of topics addressed by the contributions presented at LSO 2004 clearly reflects the interdisciplinary viewpoint required for successful knowledge management approaches for software-intensive organizations. Despite the advances reported on, further effort will need to be spent on a number of outstanding issues and challenges, in particular: Techniques, methods, and tools that allow for a lightweight, incremental phase-in of knowledge management; peer-to-peer knowledge sharing; scalability of proposed knowledge management approaches; measuring the success of these approaches, to name but a few.

Notwithstanding innovations in the domain of learning software organizations, we continue to recognize that human skills, expertise, and relationships will remain the most valuable assets of a software-intensive organization.