Knowledge Management for Learning Software Organizations

Frank Bomarius, Klaus-Dieter Althoff, Wolfgang Müller

Fraunhofer Institute for Experimental Software Engineering

Sauerwiesen 6, D-67661 Kaiserslautern

Email: {bomarius, althoff, mueller}@iese.fhg.de

Abstract

To master the challenges of globalization and tight competition, many companies are discovering management of knowledge as the strategic concept to maintain their competitiveness. In the software business, especially the increasing application complexity and frequent introduction of new technologies require optimal exploitation of available experience. In this report, we give an overview of our approach to knowledge management in software engineering and how this supports the concept of a Learning Software Organization.

Motivation for the Application of Knowledge Management

The reason behind the application of knowledge management (KM) in the software business is the need to improve software quality and reduce cost and lead time in order to remain competitive. This is a continuous endeavor that calls for a systematic approach, such as the Quality Improvement Paradigm (QIP) [BCR94a]. The QIP defines the framework for all improvement steps. But continuous improvement (within any business domain of non-trivial complexity) requires continuous learning on the organizational level. Individual learning, while still an essential requirement, is far too slow if it were the only means of learning and dissemination of knowledge: each experience made or improvement achieved in one project must be transferred as fast and efficiently as possible to other projects – this cannot be achieved by displacing project personnel (experts) without jeopardizing project success. Therefore, making knowledge explicit and managing it like a precious resource is the best alternative available.

The goals that drive KM have to be derived from the business goals of the organization to ensure maximal contribution of KM to the company's mission. KM must be organized in feedback-loops that allow monitoring of goal achievement. To allow such monitoring not only in qualitative but also in quantitative terms, goal-oriented measurement and evaluation have to be integrated with the approach. This supports learning about effects of actions based on facts rather than belief, which in turn is key to sustained (motivation for) improvement. The Goal/Question/Metric (GQM) approach [BCR94b] is a very well-suited way to come up with a suitable measurement program to gather and analyze the data required for monitoring and identifying future KM goals.

Basili and Rombach [BR88; BR91; BCR94a] have developed an organizational concept, called the Experience Factory (EF), to support the knowledge-related needs of the software business. The EF concept includes the definition of tasks (along the lines of QIP), roles, and tool support that together comprise a Learning Software Organization. In other words: KM for a Learning Software Organization is best organized according to QIP and the Experience Factory concept and is put on a quantitative basis by means of the GQM approach.

Our Approach

For each company, we develop tailored transition concepts that migrate the existing organization into an Learning Software Organization. Besides QIP, GQM, and EF, the techniques and methods employed to prepare and perform the transition include process modeling and analysis, knowledge modeling, training and further education, and introduction of information technology support.

Even though organizational learning has important managerial and socio-cultural aspects, which are addressed by training and further education, a strong support from information technology is mandatory to handle knowledge efficiently. We found that Case-Based Reasoning (CBR), a well-known approach in artificial intelligence, supports our concept in a very natural way [Alt98; TA97; TA98]. Since experts, such as senior software engineers, 'actually apply CBR' when solving problems, i.e., they recall and apply experiences from past similar projects when solving a problem at hand, CBR methods can be used to model and support major parts of their knowledge-related behavior. This is even independent from the technology employed, be it paper, a database, or a CBR tool [ABT98].

From an organizational perspective, CBR and QIP are compatible [AB+98], thus available experience on building CBR applications can be reused for building Experience Factory applications and vice versa [BA98].

A commercial CBR tool, available from a cooperation partner [Tec99], acts as a reasonable starting point for implementing the technical infrastructure for the Learning Software Organization. This tool is continuously developed further to fulfill the needs of the Learning Software Organizations.

Knowledge Formalization

In order to manage knowledge, it has to be formalized, represented (technically), and made accessible by means of search mechanisms. The conceptualization of the relevant knowledge is called ontology. We have developed a REpresentation Formalism for Software ENgineering Ontologies (REFSENO) [TG98] to describe such ontologies. REFSENO allows different degrees of formalization of knowledge from the software domain (narrative/structured text, semi-formal, formal) to be handled.

Based on REFSENO, a core ontology for modeling these kinds of knowledge has been defined [TG98]. The precise, practical, and useful definition of a core ontology for software engineering reflects one of the specific competencies of the IESE. Such competence only arises out of experimentation and the analysis of a wide spectrum of software projects. For instance, one research question we are investigating is, which context characteristics enable the effective reuse of specific software engineering technologies (e.g., inspections) [Bir97].

In software engineering we need to support knowledge acquisition from a variety of sources, namely, the software artifacts, processes, and experts [BSA98]. As of today, we use knowledge represented as process models, Goal-Question-Metric (GQM) measurement plans, project management experience, software engineering technologies, process-product relationships, and lessons learned. For each kind of knowledge, we support the appropriate knowledge life cycle [ABT98; BT98; FT98; GR+98].

Knowledge Representation

On the technical level, an ontology can be mapped to a set of data types, each being comprised of a set of attributes. This data schema could, of course, be implemented using paper, hypertext documents (e.g., HTML), or a classical relational database. To meet the different requirements of our customers and to be able to cope with the installed tool base, we have investigated several alternatives (besides CBR technology) to implement the technical support for learning organizations [GA+98]:

- Purely paper-based implementations are a good starting point in many cases, but in the long run, they will turn out to be too inefficient. While starting simple by using paper is possible, one has to be aware that soon data handling becomes too cumbersome and therefore may endanger continuation of the KM program.

- Hypertext-based solutions have the advantage that they are in widespread use and often a large amount of information is already available in enterprises in the form of hypertext documents (e.g., in Intranets). However, they have limitations concerning the versatility of the retrieval functions (namely, navigation and keyword search) and even more so regarding maintenance of knowledge.
- From the point of view of KM, databases can turn out to be very limited wrt the kinds of retrieval functions they can perform on semi-formal and informal knowledge as well as the maintenance effort they incur once a significant amount of knowledge must be handled [Alt97, GA+98].
- There are many commercial tools on the market that claim to be knowledge management tools. Most of them originated from the document management or workflow domain. We are prepared to also deploy these tools as a basis to support organizational learning in case the customer already has them in use and wants to stick with them.

Depending on the respective customer situation, it must be possible to choose one of these implementation approaches as a starting-point and build up a learning organization according to the EF principles step by step.

However, from our experience, similarity-based knowledge retrieval (which is neither readily available in an off-the-shelf database nor for paper or hypertext documents) has proven to be a powerful concept for accessing the knowledge stored in a knowledge base. Simply speaking, a CBR tool is comprised of a database and the required similarity-based retrieval functions. CBR supports our concepts quite seamlessly, and has matured to an industrial-strength technology from which we build Software Engineering Experience Environments [AB+99] jointly with the CBR tool provider.

Searching for knowledge items – or 'cases' in CBR terminology – using similarity-based retrieval functions means to specify one or more attribute values of the knowledge item(s) one is looking for in a query-by-example fashion. The CBR tool will then rank all cases of the requested type contained in the knowledge base according to their similarity to the sample query. The similarity measure is a mathematical function, basically a weighted sum of similarities between individual attributes, that has to be tuned to the needs of the user. These functions are a special kind of (compiled) knowledge about the domain.

As long as any case of the requested type is in the case base, a CBR tool will give the best answer possible according to the notion of similarity implemented in the system. Obviously, the adjustment of the similarity functions plays a central role [ANT99] and is dependent on a thoroughly selected terminology and a formalized conceptualization of the knowledge within the domain at hand. Our experience shows that this pays off very soon as the knowledge base grows. In the projects we performed it turned out that the additional effort to formally represent the knowledge was not very high compared to the overall effort required to build up a maintainable knowledge base.

Change Management Aspects

A learning organization can only thrive in a culture in which knowledge transfer is rewarded, not penalized, as can often be experienced in practice. This is easy to state, but hard and time-consuming to achieve. Considering the experience from many business reengineering projects, the introduction of a learning organization requires an overall concept that integrates socio-cultural change, process reengineering, and technology introduction. The latter of these is usually over-emphasized, neglecting the other factors.

With a learning organization culture in place, a continuous effort is necessary to nurture and grow the organizational memory that we call Experience Base (EB). Goal-oriented measurement and evaluation of the utility of the system (functionality as well as contents) [BDR96; BCR94b] can be used to guide that process and especially to re-align it with the underlying business and knowledge development goals [NT98] on a regular basis.

A sustained evaluation of the contents of the EB is mandatory to prevent it from becoming outdated, or overloaded with information no longer relevant. So, the continuous review of the knowledge in the EB is required to prevent it from degrading into a data cemetery that soon would loose credibility and, thus, would be abandoned.

Practical Experiences

The IESE and the associated Software Engineering chair at the University of Kaiserslautern have acquired significant project experience throughout the last years. The first projects dealing with the management of Software Engineering knowledge started in 1992. Additionally, there are also ten years of experience about running an Experience Factory at NASA´s Software Engineering Laboratory available through our sister institute, the Fraunhofer Center for Experimental Software Engineering, Maryland [BC+92; BM95].

The projects we have conducted where carried out with industrial partners and research institutes in Germany and Europe. These projects created solutions to capture, package, and reuse knowledge about project management, measurement programs, inspections, process-product relationships in the embedded systems domain [HJ+98], and decision support for tool selection [ANT98]. We also carried out research studies about tasks and roles in a Learning Organization, defined the processes for packaging knowledge, and investigated alternatives for knowledge modeling.

The project results and the related research activities strengthened our belief that the goal-oriented management of software engineering knowledge and the related concept of a Learning Software Organization offer considerable potential for faster, better, and cheaper software development.

Outlook

Our current challenges include the development of comprehensive experience bases (wrt types of knowledge and topics), the acquisition and tuning of similarity functions, support for context-sensitive retrieval, a consolidated procedure for the acquisition and evaluation of all kinds of relevant knowledge, procedures for experience base maintenance and evaluation, and an improved understanding of how to successfully build up experience factories at customer sites. In this way we advance our know-how and technologies to deal with the requirements of stand-alone experience factories.

On a broader level, we are also concerned with the development of a layered architecture of several interacting experience factories ranging from department level, through company level, to even cross-company experience factories. Each level has individual strategic goals, topic areas, knowledge types, and operational procedures.

References

[Alt97] Althoff, K.-D. (1997*). Evaluating case-based reasoning systems: The Inreca case study*. Postdoctoral Thesis (Habilitationsschrift), University of Kaiserslautern.

[Alt98] Althoff, K.-D. (1998). Case-Based Reasoning and Experimental Software Engineering. *Invited Talk at the 4th European Workshop on Case-Based Reasoning (EWCBR98), Dublin*, September 23-25. (slide copies available at http://demolab.iese.fhg.de:8080/Publications/ewcbr98/).

[AB+98] Althoff, K.-D., Birk, A., Gresse von Wangenheim, C. & Tautz, C. (1998). Case-based reasoning for experimental software engineering. In M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, S. Wess. (eds*.), Case-Based Reasoning Technology – From Foundations to Applications*. Springer-Verlag, 235-254.

[AB+99] Althoff, K.-D., Birk, A., Hartkopf, S., Müller, W., Surmann, D. & Tautz, C. (1999)*. Managing Software Engineering Experience for Comprehensive Reuse*. IESE-Report No. 001.99/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

[ABT98] Althoff, K.-D., Bomarius, F. & Tautz, C. (1998). Using Case-Based Reasoning for Building Learning Software Organizations. *Proc. European Conference on Artificial Intelligence 1998 Workshop on Building, Maintaining, and Using Organizational Memories (OM-98)*, Brighton, Aug. 24-28; also: http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-14/.

[ANT98] Althoff, K.-D., Nick, M. & Tautz, C. (1998). CBR-PEB: A Tool for Implementing Reuse Concepts of the Experience Factory for CBR-System Development. *To appear in Proc. 5th German Conference on Knowledge-Based Systems (XPS99) Workshop on Case-Based Reasoning (GWCBR99*); also: IESE-Report No. 058.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany. CBR-PEB is publicly accessible via http://demolab.iese.fhg.de:8080/.

[ANT99] Althoff, K.-D., Nick, M. & Tautz, C. (1999*). Improving Organizational Memories Through User Feedback.* IESE-Report No. 004.99/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

[BA98] Bergmann, R. & Althoff, K.-D. (1998). Methodology for building case-based reasoning applications. In M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, S. Wess. (eds*.), Case-Based Reasoning Technology – From Foundations to Applications*, Springer-Verlag, 299-328.

[BC+92] Basili, V.R., Caldiera, G., McGarry, F., Pajersky, R., Page, G., Waligora, S. (1992). The Software Engineering Laboratory – an operational Software Experience Factory. *Proceedings International Conference on Software Engineering (ICSE*), 370–381, May 1992.

[BCR94a] Basili, V.R., Caldiera, G. & Rombach, H.D. (1992). Experience Factory. In J. J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, 469–476. John Wiley & Sons, 1994.

[BCR94b] Basili, V.R., Caldiera, G., Rombach, H.D. (1994). Goal Question Metric Paradigm. In J. J. Marciniak, ed., *Encyclopedia of Software Engineering*, volume 1, 528–532. John Wiley & Sons, 1994.

[BDR96] Briand, L.C., Differding, C.M., Rombach, H.D. (1996). Practical guidelines for measurement-based process improvement. *Software Process*, 2(4):253–280, December 1996.

[Bir97] Birk, A. (1997). Modelling the application domains of software engineering technologies. *Proceedings Automated Software Engineering* '97, IEEE Computer Society Press, 1997.

[BM95] Basili, V.R. & McGarry, F. (1995). The Experience Factory: How to build and run one. *Tutorial given at the International Conference on Software Engineering (ICSE),* April 1995, Seattle, Washington, USA.

[BR88] Basili, V.R., Rombach, H.D. (1988). The TAME Project: Towards Improvement Oriented Software Environments. *TSE, Vol. SE-14*, No. 6, June 1988, 758-773.

[BR91] Basili, V.R., Rombach, H.D. (1991). Support for comprehensive reuse. *IEEE Software Engineering Journal*, 6(5):303–316, September 1991.

[BSA98] Birk, A., Surmann, D. & Althoff, K.-D. (1998). Applications of Knowledge Acquisition in Experimental Software Engineering. *To appear in Proc. 11th European Workshop on Knowledge Acquisition, Modeling, and Management (EKAW99)*; also: IESE-Report No. 059.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

[BT98] Birk, A. & Tautz, C. (1998). Knowledge Management of Software Engineering Lessons Learned. *Proceedings 10th International Conference of Software Engineering and Knowledge Engineering (SEKE) 1998*, San Francisco.

[FT98] Feldmann, R. & Tautz, C. (1998). Improving best practices through explicit documentation of experiences about software engineering technology. *Proceedings International Conference on Software Process Improvement in Research and Education, London, England, September 1998*. British Computer Society Quality Special Interest Group.

[GA+98] Gresse von Wangenheim, C., Althoff, K.-D., Barcia, R. M. & Tautz, C. (1998). *Evaluation of Technologies for Packaging and Reusing Software Engineering Experiences*. IESE-Report No. 055.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany.

[GR+98] Gresse von Wangenheim, C. Ramos, A.M., Althoff, K.D., Barcia, R.M., Weber, R. & Martins, R. (1998). Case-Based Reasoning Approach to Reuse of Experiential Knowledge in Software Measurement Programs. In L. Gierl & M. Lenz (eds.), *Proceedings 6th German Workshop on Case-Based Reasoning (GWCBR-98*), Technical Report, University of Rostock, IMIB series vol. 7, March 1998, 109-118; (http://www.informatik.hu-berlin.de/~cbr-ws/GWCBR98/PAPERS/Gresse.ps.gz)

[HJ+98] Hamann, D., Järvinen, J., Birk, A. & Pfahl, D. (1998). A Product-Process Dependency Definition Method. *Proc. 24th EUROMICRO Conference, Sweden*; also: IESE-Report No. 049.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern (Germany).

[NT98] Nick, M. & Tautz, C. (1998). Practical Evaluation of an Organizational Memory Using the Goal-Question-Metric Method. *To appear in Proc. 5th German Conference on Knowledge-Based Systems (XPS99)*, Springer Verlag; also: IESE-Report No. 063.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern (Germany).

[Tec99] CBR-Works (1999).(http://www.tecinno.com/english/products/cbrw_main.htm)

[TA97] Tautz, C. & Althoff, K.-D. (1997). Using case-based reasoning for reusing software knowledge. *Proceedings of the 2nd International Conference on Case-Based Reasoning*, Providence, RI, July 1997. Springer-Verlag.

[TA98] Tautz, C. & Althoff, K.-D. (1998). Operationalizing Comprehensive Software Knowledge Reuse Based on CBR Methods. In L. Gierl & M. Lenz (eds.), *Proceedings 6th German Workshop on Case-Based Reasoning (GWCBR-98),* Technical Report, University of Rostock, IMIB series vol. 7, March 1998, 89-98 (http://www.informatik.hu-berlin.de/~cbr-ws/GWCBR98/PAPERS/Tautz.ps.gz)

[TG98] Tautz, C. & Gresse von Wangenheim, C. (1998). REFSENO: A representation formalism for software engineering ontologies. To appear in *Proc. 5th German Conference on Knowledge-Based Systems (XPS99) Workshop on Knowledge Management, Organizational Memory, and Reuse*; also: IESE-Report No. 051.98/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern (Germany).