Maintaining Experience to Learn: Case Studies on Case-Based Reasoning and Experience Factory

Klaus-Dieter Althoff*, Jens Mänz* & Markus Nick**

*University of Hildesheim, Intelligent Information Systems
Marienburger Platz 22, D-31141 Hildesheim, Germany
althoff|maenz@iis.uni-hildesheim.de

**Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6, D-67661 Kaiserslautern, Germany
markus.nick@iese.fraunhofer.de

Abstract

In the past many experience factory case studies and experiments have been carried out. We summarize some development steps and research results that, from our perspective, are important. We especially focus on the integration of experience factory and case-based reasoning and report on the respective benefits and impacts of such a seamless integration for building (more) autonomous and automated knowledge-based information systems, which will be of increasing importance in the future. It is our goal to build software-agent-enacted experience factories that improve case bases using maintenance and learning methods.

Introduction

Knowledge management (KM) provides promising approaches to make software engineering (SE) more "knowledge-based". Experience factory (EF) is such an approach for which meanwhile many lessons learned are available. This contribution especially addresses the overlapping area of research of both artificial intelligence (AI) and SE. One of our goals is to enable a closer cooperation between researchers and practitioners of both fields. According to Kirn [Kir04] and Rombach [Rom04] this is a necessary and important step but requires, among others, to learn how researchers/practitioners of the respective "other" field think and work. With case-based reasoning (CBR) and EF an integration is presented that exemplifies which kind of synergies can be achieved.

It is our goal to develop information systems that have a high degree of automation and autonomy because of built-in learning capabilities. Knowledge about EF as a human-enacted learning organization is used to develop software-agent-enacted EFs that improve case bases using maintenance and learning methods. A collection of thus improved CBR systems is then used to build these future information systems.

In this contribution we focus on the integration of EF and CBR and the motivation for such future information systems.

In the following we shortly survey EF as an approach with relevance for KM in SE focusing on the above mentioned integration perspective. After a short introduction of CBR we explain the seamless integration of these AI and SE approaches that has been achieved so far. We report on the benefits and impacts of such an integration that include the use of systematic evaluation and maintenance to improve an EB together with some relevant evaluation results. Finally, we mention some still open problems including the implementation of fully automated, software-agent-enacted EFs that enable learning through maintaining experience.

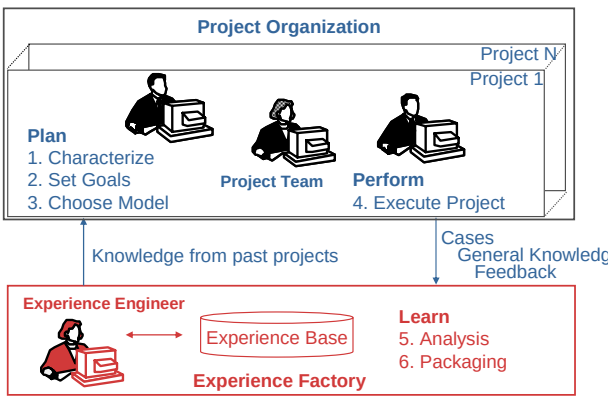

Fig. 1. Experience factory

Experience Factory

EF is a logical and/or physical infrastructure for continuous learning from experience. It includes an experience base (EB) for knowledge storage and knowledge reuse. EF is primarily an organizational approach to learning from experience (organizational learning), that is, learning is achieved by humans. There are a number of different roles and responsibilities that have been invented to support this. A characteristic feature of an EF is the separation of learning and project organization, which is motivated by practical experience from software organizations [BCR94a]. EF bases on the quality improvement paradigm (QIP), a goal-oriented learning cycle for the EB-based improvement of project planning, execution, and analysis/learning [BCR94b].

Figure 1 describes the basic "working cycle" of an EF. The project organization, shown in the upper part of the figure, is responsible for project planning and project execution. These projects provide case-specific and general knowledge including feedback. The EF then is responsible to analyze such knowledge and to package it for reuse, that is, to represent the knowledge in way that makes it useful and easy to reuse for the (projects in the) project organization.

The EF approach was introduced in the area of SE in the mid 1980s [Bas85, BR88]. . The initial early example of an operational EF is the NASA SE Laboratory (SEL) [RU89]. Meanwhile there are many EF applications in the USA as well as in Europe (e.g., [Hal96], [HSW91], [Ses96], [HSW98], [Tau00], [LR03], [SH03], [Nic05]).

These successful EF applications, especially NASA SEL, motivated further research on "learning software organizations". This includes methods for build-up and operation of an EF, the definition of the EF processes, roles, and responsibilities as well as EF implementation. The still most detailed method for EF/EB build-up including all the respective processes is described in [Tau00]. A current extension is provided by [Nic05], which includes also a systematic evaluation and maintenance support.

From an EF perspective in the mid 1990s the basic approach was already introduced by Basili, Rombach et al. With NASA SEL a very successful and established application was available. In addition, there were also some other positive examples. Important problems in the mid 1990s were how to implement an EB, how the necessary processes for developing an EF/EB should look like in detail, as well as how experiments about implementation issues could be carried out.

Case-Based Reasoning

CBR is an approach to learning and problem solving based on past experience. A past experience is stored in the form of solved problems ("cases", case-specific knowledge) in a so-called case base. A new problem is solved based on adapting solutions of known similar problems to this new problem. For solving a new problem a query is submitted to a CBR system to retrieve the solutions of the most similar problems/cases in the case base. The query is considered as a potential new case. The classical view on a case imposes that it consists of a problem description and a described solution to this problem.

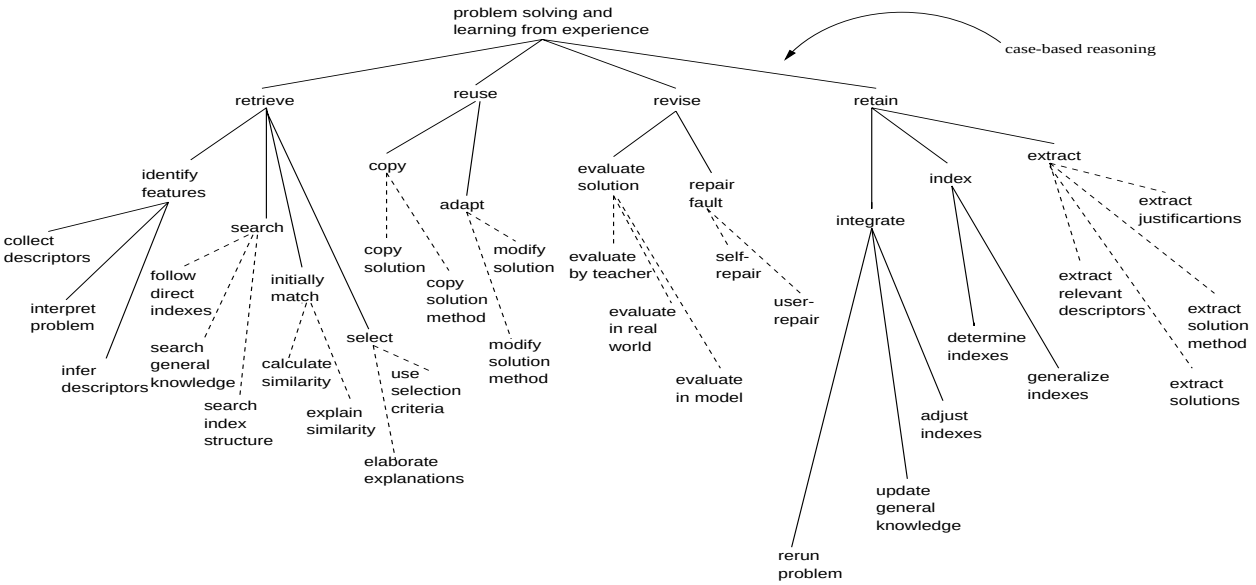

Fig. 2. CBR task-method decomposition [AP94]

Recent applications have shown that this view is too restrictive. A more general view on cases is that a case consists of a characterization (a more or less structured set of information entities) and - optionally - one or more artifacts. The characterization part is mandatory and usually contains both formal parts, which can be interpreted by the CBR system, and informal parts, which can be understood by the user only. The characterization can be completed by links to any kind of artifact. Examples for artifacts are web pages, pictures, audio-visual media, or formally described pieces of knowledge. During the retrieval a part of the formal characterization of the new case is specified. The CBR system uses this for matching

against the characterizations of the other cases in the case base. The most similar cases are retrieved and parts of their characterizations and associated artifacts are combined to yield a proposed case. Applying the suggested artifact(s) associated with the proposed case results usually in a repaired case and validated and/or revised artifact(s). The repaired case with its applied artifact(s) can be stored as a learned case, merged with previously available cases, and/or only extracted parts of it are stored. These problem solving steps are also known as the CBR cycle and/or CBR process model [AP94]. They represent the first level of the CBR task-method decomposition model [AP94], which models on a certain level of abstraction how CBR systems work. It is based on components of expertise from [Ste90]. Figure 2 shows the CBR (sub)tasks arranged in a hierarchy. The respective CBR methods either define in what order to perform the subtasks of a task or how to perform a task that is not further decomposed (see [Alt97] for a detailed description). In addition, access to knowledge is required that is needed to perform some task including both contents (domain knowledge) and pragmatic constraints imposed on this domain knowledge.

CBR was introduced in the late 1970s and early 1980s in the area of cognitive science and AI by Schank and Kolodner [Sch82, Kol93] as an approach for modeling problem solving and learning of both humans and machines. In the late 1980s and beginning 1990s this led to a focus of knowledge-based systems on experience, that is, on case-specific knowledge [AP94, Bar87, Alt89, Aha99, Alt01]. Basing on the dynamic-memory-idea of Schank guaranteed the situatedness of the approach, which often led to a good user acceptance. Accordingly a number of commercial tools and many real-life applications were developed (e.g., [Alt95, Ber03, Wat03]).

Important problems in the mid 1990s were how to systematically develop a CBR system, how to operate it, how to integrate it into an industrial environment as well as how to evaluate it.

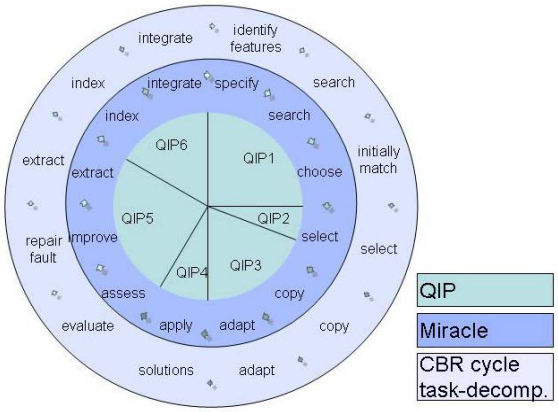

Fig. 3. Relating CBR and EF/QIP

Integrating Case-Based Reasoning and Experience Factory

Since both approaches EF and CBR deal with learning from experience they are obviously related. To our knowledge the first who mentioned that CBR could be used for EF was Henninger [Hen95]. However, the key idea how to seamlessly integrate EF and CBR was first mentioned by Althoff and Wilke, and then further detailed by Tautz and Althoff [TA97, TA98, Alt98, Tau00]. The integration idea between EF and CBR is very simple but, nevertheless, was also very helpful: Interpret the methods within the CBR task-method decomposition model organizationally. That is, in the first place consider them as tasks to be done by some human being, and which maybe partially supported by a software system. One important aspect here is that CBR, as originally being introduced for modeling human problem solving and learning, can also be carried out by human beings. So, in principle it was possible to enact the CBR task-method decomposition model by human beings. As a consequence, CBR and EF/QIP became comparable (see Fig. 3). By this, the detailed knowledge on CBR processes elaborated in the CBR community provided a very good starting point for describing the EF processes ([AP94], [AA96], [Alt97]). Thus, human-based execution of tasks was the natural means to integrate CBR and EF/QIP. With components of expertise a framework for developing knowledge-based systems – here instantiated for CBR systems – was used for carrying out organizational learning. Of course, CBR was a first natural implementation technology for this framework, that is, could easily be used to automatically support the execution of the one or the other (sub)task.

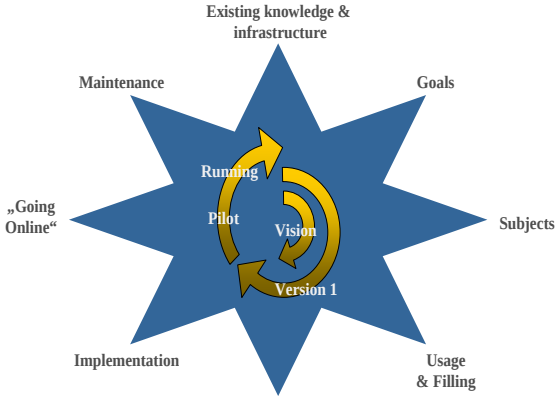

Fig. 4. Goal-oriented CBR/EF systems

For using this framework, Tautz [Tau00] developed the DISER method, a systematic, detailed method for building integrated CBR/EF systems supporting both human-based and computer-based task execution. Figure 4 shows the four main development phases (vision, version 1, pilot, running) as well as the main aspects of developing a CBR/EF system: Goals/existing knowledge and infrastructure, subjects, usage and filling, modeling, implementation, "going online", and maintenance. Based on its EF part, DISER provided knowledge about how to organizationally embed an EF/CBR system in commercial environments. This included the EF roles that base on experiences with tasks and responsibilities from existing EF approaches:

- The *manager* provides resources, defines strategic goals and initiates improvement programs. He determines the structure and content of the case base and controls its quality.
- The *supporter* is responsible for documenting new experiences and supporting the project team. He collects and qualifies artifacts from the projects in accordance with the reuse criteria and the goals of the engineer. Upon request, he supports the project team in retrieving and modifying the experience knowledge.
- The *engineer* is responsible for packaging and analyzing existing experiences. Together with the manager, he identifies new reuse criteria and, based on that, acquires new cases. He analyzes the case base in order to detect (further) improvement potential.
- The *librarian* is responsible for technical aspects like setting-up and maintaining the case base, storing, and publishing new cases.

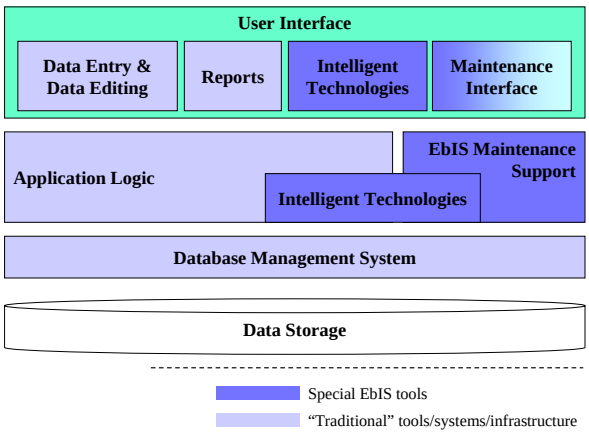

Fig. 5. Product-line architecture for EF/EB systems

Again based on its EF part, DISER also included goal-oriented measurement and evaluation, an approach that is easily usable for evaluating CBR systems [NAT99, NF00]. From an AI perspective, the use of a systematic SE method like DISER led to an extension of the technology domain, that is, besides CBR also other search and repository technologies became usable. This new technology domain is called "Experience-based Information Systems" (EbIS). An EbIS is an information system that includes experiences (case-specific knowledge). In addition, the potential use of other (than CBR) AI technologies was identified [e.g., text/data mining: RA04]. Some of the mentioned advantages have meanwhile been confirmed by other research groups through the use of CBR in EF/SE as well as EF in CBR/AI (e.g., [Hen95], [She03], [Gom04] as well as [Kal00], [Bar02], [Ber03]).

From a SE perspective, the use of mature technology like CBR resulted in the introduction of systematic reuse into the EB development process, namely the software product-line approach based on some product-line architecture [Nic05] (see Fig. 5). Based on CBR technology a number of projects were carried out that brought to light some deficiencies of DISER concerning [Nic05]:

- phase models and development strategies for a better integrability on the software process side;
- solutions for "feedback loops" as well as experience life cycle models;
- solutions for relating different types of knowledge/experience each represented on a different level of granularity;
- rapid application development approaches for a „cheap start";
- knowledge modeling approaches/guidelines for "scaling up";
- scalability of the underlying knowledge technology;
- integrability of knowledge technology with traditional software system technology;
- supporting maintenance as a knowledge-intensive task;
- maintenance process;
- decision support for maintenance;
- acquisition method for maintenance knowledge;
- maintenance enactment support (for optimizing the maintenance process);
- business goal oriented method for running an EbIS;
- relating maintenance to the goals of an EbIS to guide maintenance with evaluation;

- availability of an evaluation plan and maintenance knowledge already for the beginning of regular use for handling the to be expected continuous stream of experience.

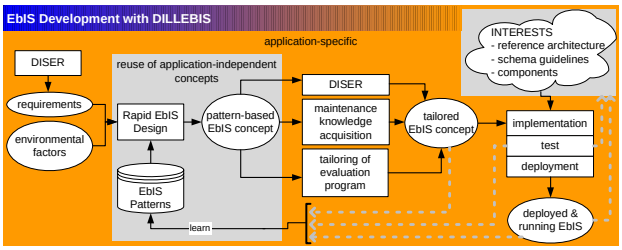

Fig. 6. EbIS development process with DILLEBIS [Nic05]

Based on numerous projects the DISER successor DILLEBIS [Nic05] has overcome these deficiencies. Figure 6 describes, on a certain level of abstraction, how an EbIS can be developed using DILLEBIS. Based on learned "development patterns" rapid application development becomes possible. The knowledge acquired through the continuously running evaluation is directly used for maintenance [Nic05]. Figure 7 details this knowledge capturing process: In parallel to the deployed system a monitoring evaluation program is running. Then both is supporting maintenance the knowledge gathered for preparing evaluation and the achieved evaluation results. The deployed EbIS is developed based on the resulting tailored EbIS concept and the available reference architecture, schema guidelines, and components.

The main advantages of DILLEBIS include its integration into the respective work processes, feedback-based learning, evaluation-guided maintenance (see Fig. 7) as well as enabling a low-cost start in customer projects using EbIS patterns and a product-line architecture (see Fig. 5). The product-line architecture helped to introduce a number of variabilities with respect to search, storage, and maintenance techniques.

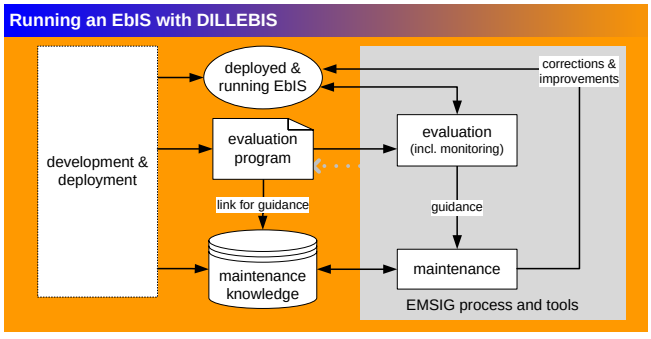

Fig. 7. Running an EbIS with DILLEBIS [Nic05]

Benefits

Integrating EF and CBR resulted in various advantages. DISER/DILLEBIS generalized from CBR Systems to EbIS, a newly defined type of information system that contains experiences and is able to handle a (more or less) continuous „stream of experience".

DISER/DILLEBIS explicitly considers that an EbIS is embedded into a "lived" process. Here DISER/DILLEBIS goes beyond other methods for CBR system development/maintenance (e.g., Inreca [Ber99], SIAM [RB00]). Such approaches strictly stay within the "CBR domain". They do not open up for other AI technologies, and do not explicitly consider the task of embedding the developed software system into the respective business processes (e.g., maintenance approaches remain on the technical level only). DISER/DILLEBIS treats this business process with much more precision than these other CBR system development methods do. In contrast to other CBR approaches that mainly use classification accuracy oriented evaluation approaches from machine learning, DISER/DILLEBIS includes a goal-oriented measuring and evaluation approach for knowledge bases that is more flexible and more appropriate for evaluating complete software systems/components [NAT99, NF00, BCR94b].

Since DISER/DILLEBIS generalized from the "CBR domain" to the "EbIS domain", one natural question is which technologies can contribute to EbIS development and how? One idea is using supplementing AI technologies that naturally can extend CBR, e.g., using machine learning techniques as part of the DISER method [Rec02]. Another idea is using other AI technologies that naturally can substitute CBR, e.g., ontology based knowledge modeling and retrieval, model based reasoning, and/or information retrieval.

Evaluation

In [Tau00] an experiment was carried out that proved that DISER basically worked. The EbIS approach (»using the EbIS«) was scientifically compared with the human-based approach (»talking to colleagues«).

The experiment focused on a typical scenario within a Fraunhofer institute. Someone is responsible for a new customer project and has to collect all important information about the customer as well as about similar projects and/or customers.

The experiment showed:

- *Efficiency*: The EbIS approach finds more useful guidelines and observations per time period (in terms of both effort and duration).

- *Effectivity*: The EbIS approach finds useful guidelines and observations not obtained by the human-based approach.

The experiment validated this in a statistically significant way. Thus, the result of the experiment was to combine the human-based and the EbIS approach. The participants in the experiment agreed: 28 out of 29 participants would apply both approaches in combination.

In the scope of [Nic05] a series of case studies was carried out where it was analyzed how well the modeling related parts of DISER/DILLEBIS were, especially with respect to maintenance. The goal was to evaluate the initial concepts of the EbIS development method. Two role playing games were carried out with seven students each with nearly identical design[1]. The context was a company for developing CBR applications called X-CBR that is a few years old and can be viewed as an SME.

The task was following: X-CBR management decided that an EF on CBR projects should be developed, because this would enable X-CBR to better manage their knowledge on their core competence „CBR applications".

Lecturers and students took over the roles of the EF and the X-CBR organization. Within the lecture the initial modeling of the EbIS was carried out. At the end of the lecture(s) a final questionnaire had to be filled out. In addition, a feedback meeting was carried out.

The following results were achieved:

- The initial modeling was completed within 180 min (2 lectures of 90 min each)
- The case study provided a lot of valuable experience for an initial workshop with a financial services company.
- During the case study it came out that DISER/DILLEBIS needed a phase concept which was introduced directly after the case study.

Further evaluation results can be found in [Tau00], [Nic05], and [Rec04]. The following overview in Figure 8 on relevant EbIS application projects is due to [Nic05].

Status	**#**	***Size of EbIS** (#User, #Cases)*		
		Small	*Medium*	*Large*
Successful	**8**	SKe-Pilot, ISI, SLI-EB, IPQM, KM-PEB, CBR-PEB	indiGo/ CoIN	ESERNET
Implementation	**1**			T-Com SR
Status unknown	**1**		Project A	
#successful		**6**	**1**	**1(+1)**

Fig. 8. Overview on EbIS projects [Nic05]

The above mentioned EbIS methods showed their broad applicability through successfully occurring in numerous real-life projects. The listed projects are different in size and project type and thus show a certain "breadth". An EbIS is considered to be successful if it is in usage for more than one year (status "accepted") or its status is "deployable" and it is tightly integrated. An assumption for tight integration is the acceptance and correct usage of the tool that supports the business process.

Impact and Outlook: Planned Research Activities

The authors are convinced that the integration of SE and AI methods and techniques will be a very important future research task, and that maintaining a case/experience base is an important example for this.

We shortly reported about the integration of CBR and EF and about the application of software product-lines to EF/CBR systems (EbIS). The authors believe that software product-lines and EF/CBR have to be seamlessly integrated with a specific focus on the knowledge aspects. The more general case here is to integrate software product-lines with knowledge-based systems, which leads to what we call a "knowledge-line" ("Wissenslinie")[2]. We plan to realize software architectures for such knowledge-lines based multi-agent system technology because this allows to have the flexibility that will be required for future information systems. CBR/EF will be used for implementing the respective agents, which leads to a hierarchical, mixed software-agent-/human-based EF, where each agent is embedded in a "digital EF" and the multi-agent-based software product-line will be an EF of (digital) EFs. Such agents will learn from experience that is collected in their case base. Other software agents that take over the classical EF roles, continuously try to improve the content of the case base, for example adding/deleting new cases, improving the similarity assessment, or generating new/improved adaptation rules.

One major idea behind this approach is using knowledge management processes as a kind of specification of "learning processes". By this we mean processes that contribute to achieve a higher degree of autonomy and automation through experience based improvement. These are then implemented as software-agent-enacted EF processes. Thus, for each software agent the EF processes would be completely automated.

How to integrate more knowledge into software product-lines and their supporting processes has been addressed by various authors [DM05, Sch04, Mut05, Bay05]. Some first ideas about knowledge-lines can be found in [DA04]. An example for using multi-agent system technology for CBR is given by [Pla05].

[1] Further role playing games took place with similar design in other lectures/semesters.

[2] For a motivation of such an integration see, for example, [Bib04] and/or [RA04].

References

[AA96] Althoff, K.-D. & Aamodt, A.: Relating case-based problem solving and learning methods to task and domain characteristics: towards an analytic framework. *AI Communications, 9 (3), 1996*, 1-8.

[Alt98] K.-D. Althoff, A. Birk, C. Gresse von Wangenheim, and C. Tautz. Case-Based Reasoning for Experimental Software Engineering. In: M. Lenz, B. Bartsch-Spörl, H. D. Burkhard & S. Wess, editors, *Case-Based Reasoning Technology – From Foundations to Applications*. Springer Verlag, 1998, 235-254.

[Alt97] Althoff, K.-D.: *Evaluating Case-Based Reasoning Systems: The INRECA Case Study*. Habilitationsschrift, Fachbereich Informatik, TU Kaiserslautern, 1997.

[AP94] Aamodt, A. & Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications 7(1)*, 1994, 39-59.

[Aha99] Aha, D.W.: The AAAI-99 KM/CBR Workshop: Summary of Contributions. *Proc. ICCBR ´99 Workshops, II-37–II-44*. Technical Report, LSA-99-03E, University of Kaiserslautern, 1999.

[Alt89] Althoff, K.-D., Kockskämper, S., Maurer, F., Stadler, M. and Wess, S.: Ein System zur fallbasierten Wissensverarbeitung in technischen Diagnosesituationen. In: Retti, J. and Leidlmeier, K. (eds.), *5th Austrian AI-Conference*, Springer Verlag, 1989; 65-70.

[Alt95] Althoff, K.-D., Auriol, E., Barletta, R. & Manago, M. (1995). *A Review of Industrial Case-Based Reasoning Tools*. AI Perspectives Report, Oxford, UK: AI Intelligence.

[Alt00] Althoff, K.-D., Birk, A., Hartkopf, S., Müller, W., Nick, M., Surmann, D. & Tautz, C.: Systematic Population, Utilization, and Maintenance of a Repository for Comprehensive Reuse. In G. Ruhe et al. (eds.), *Learning Software Organizations - Methodology and Applications*, Springer Verlag, LNCS 1756, 2000; 25-50.

[Alt01] Althoff, K.-D.: Case-Based Reasoning. In: S.K. Chang (Ed.), Handbook on Software Engineering and Knowledge Engineering. Vol.1, World Scientific, 2001; S. 549-587.

[AW97] Althoff, K.-D. & Wilke, W.: Potential Uses of Case-Based Reasoning in Experience Based Construction of Software Systems and Business Process Support. In: R. Bergmann & W. Wilke (eds.), *Proc. 5th German Workshop on Case-Based Reasoning (GWCBR'97)*, LSA-97-01E, Centre for Learning Systems and Applications, University of Kaiserslautern, 1997, 31-38.

[Bar87] Bartsch-Spörl, B.: Ansätze zur Behandlung von fallorientiertem Erfahrungswissen in Expertensystemen. *KI 4(1987)*, 32-36.

[Bar02] Bartlmae, K.: Die KDD-Experience Factory: Ein Unterstützungsansatz für die Wissensentdeckung in Datenbanken. Jena, Germany, Wirtschaftswissenschaftliche Fakultät der Friedrich-Schiller-Universität, Dissertation, Februar 2002.

[Bas85] Basili, V.R.: Quantitative evaluation of software methodology. In Proceedings of the First Pan-Pacific Computer Conference, Melbourne, Australia, September 1985.

[Bay05] Bayer, J.: Aufbau und Einsatz einer Erfahrungsdatenbank zur systematischen und organisationsweiten Verbesserung von Software-Dokumentation. *KI (3)2005*, 70-72.

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

[BCR94b] Basili, V.R., Caldiera, G. & Rombach, H.D.: Goal Question Metric Paradigm. In Marciniak, J.J. (ed.), Encyclopedia of SE, vol 1, Wiley & Sons, 1994; 528-532.

[BR88] Basili, V.R. & Rombach, H.D.: The TAME Project: Towards improvement-oriented software environments. IEEE *Transactions on SE, SE-14(6), 1988*; 758-773.

[Ber03] Bergmann, R., Althoff, K.-D., Breen, S., Göker, M., Manago, M., Traphöner, R. & Wess, S.: Developing Industrial CBR Applications. Springer Verlag, LNAI 1612, 2003

[Ber99] Bergmann, R., Breen, S., Göker, M., Manago, M. & Wess, S.: *Developing Industrial CBR Applications: The INRECA Methodology*. LNAI 1612, Springer, 1999.

[Bib04] Bibel, W., Andler, D., Da Costa, O., Küppers, G. & Pearson, I. D. (2004). Converging Technolo¬gies and the Natural, Social and Cultural World. Report of the EU High Level Expert Group on „Forsighting the New technology Wave“, 30.6.2004.

[BR00] Broomé, M. & Runeson, P.: Technical requirements for the implementation of an experience base. In G. Ruhe et al. (eds.), *Learning Software Organizations - Methodology and Applications*, Springer Verlag, LNCS 1756, 2000;S. 87-102.

[DA04] Decker, B. & Althoff, K.-D.: Prozesslernen und Erfahrungsmanagement: Ergebnisse aus dem indiGo-Projekt. In: *Proc. Lernen - Wissensentdeckung – Adaptivität 2004*, 138-145

[DM05] Decker, B. & Muthig, D.: A community based approach for organizing software product line evolution. In K.-D. Althoff et al. (eds.), WM2005: Professional Knowledge Management – Experiences and Visions, Kaiserslautern: DFKI GmbH, 2005, 62-66.

[Din00] Dingsoyr, T.: An Evaluation of Research on Experience Factory. In K.-D. Althoff et al. (eds.), *Learning Software Organizations. Proc. 2nd Internat. Workshop (LSO'00)*; 55-66.

[Gom04] Gomes, P.: *A Case-Based Approach to Software Design*. Ph.D thesis, Departamento de Engenharia Informática - Universidade de Coimbra, Universidade de Coimbra, 2004.

[Hal96] Haley, T.J.: Software process improvement at Raytheon. *IEEE Software 13(6), 1996*; 33-41.

[Hen95] Henninger, S.: Developing domain knowledge through the reuse of project experiences. In M. Samadzadeh (ed.), Proc. Symposium of Software Reusability SSR 1995, 186-195.

[HSW98] Houdek, F., Schneider, K. & Wieser, E.: Establishing experience factories at Daimler-Benz: An experience report. *Proc. 20th Internat. Conf. on SE (ICSE'98*).

[HSW91] Humphrey, W.S., Snyder, T.R. & Willis, R.R.: Software process improvement at Hughes Aircraft. *IEEE Software 8, 1991*; S. 11-23.

[Kal00] Kalfoglou,Y.: On the convergence of core technologies for knowledge management and organisational memories: ontologies and experience factories. *Proc. ECAI 2000 Workshop on Knowledge Management and Organisational Memories, Berlin, 2000*, 48-55.

[KCS01] Kamel, A., Chandra, M. & Sorenson, P.G.: Building an Experience Base for Product-line Software Development Process. In R. Weber & C. Gresse v. Wangenheim (eds.), *Workshops at 4th Internat. Conf. on CBR*, 2001.

[Kir04] Kirn, S.: Interview mit Herrn Prof. S. Kirn, Universität Hohenheim. KI (3)2004, 39-40.

[Kol93] Kolodner, J.L.: Case-Based Reasoning. Morgan Kaufmann Publishers, San Mateo, 1993

[LR03] Lindvall M. & Rus I.: Lessons Learned from Implementing Experience Factories in Software Organizations. In Proc. Workshop on Learning Software Organizations 2003, Part of WM2003: Professionelles Wissensmanagement - Erfahrungen und Visionen, GI-Edition Lecture Notes in Informatics, 2003, 59-63.

[Mut05] Muthig, D.: Systematischer Aufbau und Einsatz von Wissen zur effizienten Entwicklung von Software-Varianten. *KI (2)2005*, 5-11.

[NAT99] Nick, M., Althoff. K.-D. & Tautz, C.: Facilitating the Practical Evaluation of Knowledge-Based Systems and Organizational Memories Using the Goal-Question-Metric Technique. Proc. Banff Workshop on Knowledge Acquisition, Modeling, and Management (KAW'99), 1999.

[NF00] Nick, M. & Feldmann, R.: Guidelines for evaluation and improvement of reuse and experience repository systems throught measurement programs. Proc. 3rd European Software Measurement Conference (FESMA-AEMES 2000), Madrid, Spain, 2000.

[Nic05] Nick, M.: Experience Maintenance through Closed-Loop Feedback. Dissertation, Fachbereich Informatik, TU Kaiserslautern (eingereicht), 2005.

[Pla05] Plaza, E.: Cooperative Reuse for Compositional Cases in Multi-Agent Systems. *Proc. 6th Internat. Conf. on Case-Based Reasoning (ICCBR 2005)*, Chicago, 2005.

[RA04] Rech, J. & Althoff, K.-D.: Artificial Intelligence and Software Engineering - Status and Future Trends. *Themenschwerpunkt KI & SE, KI (3)2004*, 2004, 5-11.

[RB00] Roth-Berghofer, Th.: *Knowledge maintenance of case-based reasoning systems. The SIAM methodology*. Akademische Verlagsgesellschaft Aka GmbH, DISKI 262, Berlin. Ph.D. thesis. April 2003.

[Rec02] Rech, J., Althoff, K.-D., Decker, B., Klotz, A., Leopold, E. & Voss, A.: Thoughts on Text Mining in Organizational Process Learning. *Proc. 15th German Workshop on Machine Learning (FGML'02)*, GI-Workshop-Woche "Lernen - Lehren - Adaptivität (LLA'02)", Universität Hannover, 7.-11. Okt. 2002.

[Rec04] Rech, J., Decker, B., Klotz, A., Leopold, E., Althoff, K.-D. & Voss, A.: Abschlussbericht indiGo, IESE-Report 076.04/D, Kaiserslautern, 2004.

[Rom04] Rombach, H.D.: Interview mit Herrn Prof. D. Rombach, Fraunhofer IESE und TU Kaiserslautern. KI (3)2004, Themenschwerpunkt KI und SE, 36-38.

[RU89] Rombach, H.D. & Ulery, B.D.: Establishing a measurement based maintenance improvement program: Lessons learned in the SEL. Proc. of the Conference on Software Maintenance, IEEE Computer Society Press, October 1989, S. 50-57.

[Sch82] Schank, R. C.: Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University Press, 1982.

[Sch04] Schmid, K.: Systematische Wiederverwendung im Produktlinienumfeld – Ein Enscheidungsproblem. *KI (3)2004, Special Issue on AI and SE*, 33-35.

[She03] Shepperd, M.: Case-Based Reasoning and Software Engineering. In A. Aurum, R. Jeffery, C. Wohlin & M. Handzic (eds.), Managing SE Knowledge, Springer, 2003; 181-198.

[SH03] Schneider, K. & von Hunnius, J.: Effective Experience Repositories for Software Engineering. Proc. International Conference on SE (ICSE 2003), Portland/Oregon, 2003; 534

[Ses96] Seshagiri, G.: Continuous process improvement: Why wait till level 5? Proc. 29th Hawaii Internat. Conf. on System Sciences, IEEE Computer Society Press, 1996; 681-692.

[Ste90] Steels, L. (1990). Components of Expertise. *AI Magazine, vol.11, nr. 2*, 28-49, Menlo Park, CA: The AAAI Press.

[TA97] Tautz, C. & Althoff, K.-D.: Using Case-Based Reasoning for Reusing Software Knowledge. Case-Based Reasoning Research and Development. 2nd International Conference (ICCBR'97), Providence, RI, Springer Verlag, 1997.

[TA98] Tautz, C. & Althoff, K.-D.: Operationalizing Comprehensive Software Knowledge Reuse Based on CBR Methods. In L. Gierl & M. Lenz (eds.), *Proc. of the 6th German Workshop on Case-Based Reasoning (GWCBR-98)*, Technical Report, University of Rostock, IMIB series vol. 7, March 1998, 89-98.

[Tau00] Tautz, C.: Customizing Software Engineering Experience Management Systems to Organizational Needs. Dissertation, FB Informatik, TU Kaiserslautern; Fraunhofer IRB Verlag, 2000

[Wat03] Watson, I. (ed.): Applying Knowledge Management: techniques for building corporate memories. Morgan Kaufmann Publishers Inc. San Francisco CA, 2003