Fraunhofer Institut
Experimentelles
Software Engineering

Experience Management: The Fraunhofer IESE Experience Factory

Authors:
Klaus-Dieter Althoff
Björn Decker
Susanne Hartkopf
Andreas Jedlitschka
Markus Nick
Jörg Rech

Accepted for publication in
Proceedings of Industrial Conference on
Data Mining, Leipzig, July 24-25, 2001
P. Perner (ed.)

IESE-Report No. 035.01/E
Version 1.0
July 2001

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competetive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
D-67661 Kaiserslautern

Abstract

Experience Management (EM) is an area that is increasingly gaining importance. Its roots lie in Experimental Software Engineering ("Experience Factory"), in Artificial Intelligence ("Case-Based Reasoning"), and in Knowledge Management. EM is comprised of the dimensions methodology, technical realization, organization, and management. It includes technologies, methods, and tools for identifying, collecting, documenting, packaging, storing, generalizing, reusing, adapting, and evaluating experience knowledge, as well as for development, improvement, and execution of all knowledge-related processes. The main difference between experience knowledge and general knowledge is the fact that normally, a (more or less) continuous "stream of knowledge" must be processed. Within this paper, we present some basic methods of EM, using the Fraunhofer IESE Experience Factory as an example, which, after a one-year trial run, has been in regular operation since the beginning of this year.

Table of Contents

1 Introduction

In all emerging areas of business and engineering science, there is normally a lack of explicit knowledge about their underlying processes, products, and technologies. Usually, such knowledge is built up through individual learning from the experience of the people involved. The area of organizational learning, as one part of knowledge management, tries to increase the effectiveness of individual human learning for the whole organization. Besides improving internal communication (group learning) [GV+01], organizational learning also includes documenting relevant knowledge and storing it (for reuse) in an organizational, corporate memory [AB+98, vH+96]. The learning target for a learning organization is to enable its members to effectively quarrel situational requirements taking past experience into account. Providing a higher number of alternative decisions or proceedings to employees than they would have had based on their individual repertoire characterizes a learning organization [Klu99].

An approach known from software engineering called Experience Factory (EF) [BCR94] goes one step further. Knowledge (in the form of processes, products, and technologies) is enriched by explicitly documented experience (e.g., lessons that were learned during the practical application of the knowledge). The EF approach includes capturing, documenting, storing, and disseminating of such experience. These "experience packages" are stored in an experience base (EB), which is an organizational memory for relevant knowledge and experience. The EF approach tries to explicitly rebuild human "learning from experience" to further support organizational learning. EF has to be supplemented on a technical system implementation level to realize the EB. One specific technology form artificial intelligence, which has its roots in the knowledge-based systems as well as in the machine learning subfield, is case-based reasoning (CBR) [Kol93]. We call the research area dealing with both the organizational and the technical support for "learning from experience" Experience Management (EM), which is in accordance with similar suggestions from literature [Tau00, Ber01].

EM deals with the identification, storage, and re-/use of multifaceted knowledge of the members of an organization, who acquired this knowledge through learning from a (more or less) continuous stream of experience.

Within this paper we first (Sec. 2) give an introduction to EM including case-based reasoning and EF followed by the presentation of an example for an operative EF (Sec. 3) and a method for building an EF (Sec.4). After presenting the content framework (Sec. 5) we are detailing some features of EF like business

processes and lessons learned (Sec. 6) and looking towards the aspects of maintenance (Sec. 7). Enhancing the presented approach we introduce new strategies to capture, process, disseminate, and exchange knowledge (Sec. 8) and have a look to data mining in the context of EM (sec. 9). Finally, a short summary is given (Sec. 10).

2 Experience Management

Experience management defines and develops methods for structuring and dealing with experience of experts in a particular subject, and it is becoming an increasingly important domain of knowledge management. Software engineering is a highly dynamic field in terms of research and knowledge, and it depends heavily upon the experience of experts for the development and advancement of its methods, tools and techniques. For example, the tendency to define and describe "best practices" or "learned lessons" is quite distinctive in the literature [BT98; AW00; TAN00].

In Software Engineering, the EF approach was introduced in the late eighties [BCR94; AB+00]. It explicitly deals with continuous (organizational) learning from experience. In the areas of Cognitive Science and Artificial Intelligence, CBR emerged in the beginning eighties as a model for human problem solving and learning [Sch82]. In Artificial Intelligence, this led to a focus of knowledge-based systems on experience (experience knowledge, case-specific knowledge) in the late eighties and beginning nineties, mostly in the form of problem-solution cases [BaS87; AK+89; Aha99].

In the eighties and nineties, various approaches in economical and social science as well as in business information systems, which explicitly dealt with knowledge as a resource of increasing importance, merged under the notion of knowledge management [Rom98; Leh00]. In spite of the high number of approaches and their heterogeneity, two main categories can be identified [ADK98; BJA01]: On the one hand, there are process-oriented approaches, which base mainly on communication and collaboration [JH+00], on the other hand, product-oriented approaches, which base on documentation, storage, and reuse of enterprise knowledge [AB+00]. While the former use techniques from computer supported collaborative work and workflow management, the latter build on information technology tools for documenting knowledge: Database systems, repository systems, hypertext systems, document management systems, process modeling systems, knowledge-based systems, case-based reasoning systems, etc. [Goo99].

From a more general perspective it can be stated that product- and process-oriented approaches are still not integrated. Usually they are used independently from each other, or as alternatives. As one exception here, meanwhile a deep – that is the cognitive science foundations considering - integration of the approaches of EF and CBR has been achieved [Tau00; Alt01].

2.1 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") in a so-called case base. A new problem is solved based on adapting solutions of known similar problems (see Figure 1) to this new problem.

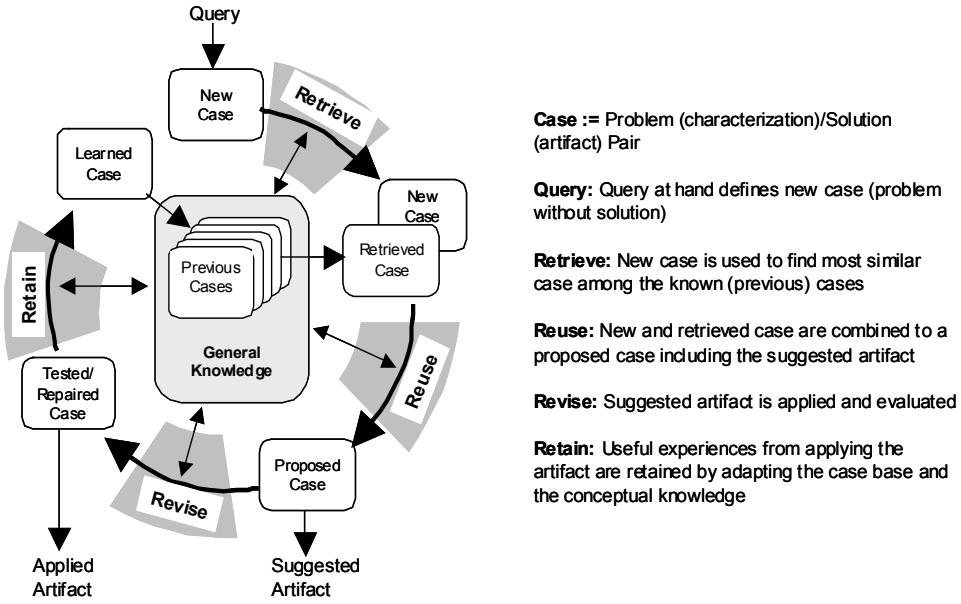

Figure 1 Case-Based Reasoning Process Model

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. Recent applications have shown that this view is too restrictive [Bur98]. 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 [ABT98, ABT00]. 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].

CBR has been used to create numerous applications in a wide range of domains including financial analysis, risk assessment, technical maintenance, process control, quality control, medical diagnosis, software support systems, forecasting, planning, design, classification of objects, photo-interpretation, real estate appraisal, electronic commerce, customer support, knowledge management, software engineering, etc. [AA+95, ABS96, BB+99, Ber99].

The underlying idea of CBR is simple: Do not solve problems from scratch but remember how you (or someone else) solved a similar problem and apply this knowledge to solve your current problem. The notion of CBR was introduced by Roger Schank. The motivation of his work was based on a cognitive science perspective and one early application area was the cognitive oriented research on story understanding. Another root of current CBR research was the motivation to use knowledge-based methods for legal reasoning, initiated by Edwina Rissland [Ris83]. A third root for current CBR might be found in research on analogical reasoning (e.g., [GH80]), though early work on CBR and analogical reasoning were quite independent from each other for many years.

After more then twenty years of existence "modern CBR" still consists of work done in the areas mentioned above. However, it has also become an engineering discipline for using knowledge and software engineering methods to develop real-life applications. In the beginning of the 90's the first commercial CBR tools appeared and, since then, more mature tools have been developed [AA+95, SW98]. While for some years the strong focus on various real-life applications seemed to separate current CBR research and development activities from its origins, in the most recent years methods and ideas from the more humanities oriented parts of CBR again become very important. Complex application domains like knowledge management, education and training, and continuous quality improvement require broadly applicable techniques. It is one of the biggest challenges of current CBR research and development to not only develop solutions for very specific (small) problems but to also develop more general solution strategies that allow, for instance, an integration of computer and human based methods.

2.2 Experience Factory

EF is a logical and/or physical infrastructure for continuous learning from experience, including the experience base (EB) for the storage and reuse of knowledge. The EF approach was invented in the area of software engineering

in the mid eighties [Bas85, BR88, BR91, BCC92, BCR94]. As practice shows, it is substantial for the support of organizational learning, that the project organization and the learning organization are separated [BA+94]. This is the main feature of the EF. It is based on the Quality Improvement Paradigm, which is a goal-oriented learning cycle for experience based improvement and evaluation of project planning, project execution and project analysis. [BCR94].

The initial example for an operating EF is the NASA Software Engineering Laboratory (SEL) [RU89; BCC92; MP90]. In the meantime EF applications were developed in the USA and also in Europe [Hal96; HSW91; Ses96; Rom96; HSW98; TA00]. The great amount of successful EF applications gave the ignition to study "Learning Software Organizations" more intensive regarding the methodology for building up and running an EF. This also includes the definition of according processes, roles and responsibilities, and last but not least the technical realization [BR00, CD00; Din00, AB+00]. The most detailed methodology for the build-up of an EF/EB on project knowledge also for the presentation of the according processes is given in [Tau00]. Since the mid nineties CBR is used both on the organizational EF process level as well as the technical EB implementation level [Hen95; AW97; TA97]. Meanwhile this approach establishes itself more and more [BB+99; KM+00; Alt01]. EF is increasingly emerging towards a generic approach for experience management as an organizational structure for reuse of knowledge and especially experience. This includes also applications independent from the software engineering domain. For example, supporting the continuous improvement process in hospitals [AB+99], the field of help-desk and service support [SW98], and the management of "non-software"-projects [BE+01]. Future trends in the scope of EF include the detailing of all necessary policies, validation and empirical evaluation [CD00; Tau00; BSL99], gaining experience with the technical realization of huge EFs, integration with the according business processes [DJ+01, JA+01], and the running of EFs [NAT01].

3 IESE Corporate Information Network: The Experience Factory

The EF deals with the typical problem that the main experience of an organization resides in the brains of some few experts. With the fast growing of our Institute this problem strengthend, because these experts are rarely accessible because of their involvement in many different tasks. Therefore, this small group of experts becomes a scarce resource as information providers. Hence, it is important (a) to provide the less experienced people with default processes and guidelines to jump-start them and (b) to facilitate experience sharing among them to build up their expertise more quickly. Since the size of our institute does not allow to talk to all people on a weekly basis, experience sharing on a personal basis does not work. Therefore, a project named COIN (Corporate Information Network) was launched. Additionally, COIN is used as a real project environment for the development and validation of technologies and methods for goal-oriented experience management including knowledge elicitation, processing, dissemination, presentation, maintenance, and evaluation. It consists of three main parts: the EB, the COIN team, and an intranet representation.

Within the EB included in COIN, all kinds of experience necessary for our daily business are stored (e.g., projects, business processes, document templates, guidelines, observations, improvement suggestions, problems that occurred and problem fixes that were applied). Defined processes (structured interviews within project touch-down meetings, see Sec. 6.1) populate this EB systematically with experience typically needed by our project teams. Dedicated improvement processes analyze problems that have occurred, devise improvement actions to avoid their recurrence, and implement strategic decisions by the institute's leadership. However, elicitation, distribution, and integration of process descriptions and lessons learned need an investment of effort [DA+01]. The project teams using the process descriptions and gaining the experiences cannot be expected to invest this effort. Compared to the objectives of the organization, projects have a short-term perspective, focusing on the development goals of the project. Therefore, an organizational unit, which is responsible for knowledge management is required. This organizational unit has to be separated from the project teams. According to [BCR94, ABT00], this separate organizational unit is called EF, which for the IESE is run by the COIN team [Tau00].

The current focus is on two major subject areas: business process descriptions and lessons learned. The lessons learned are in the form of guidelines, observations, and problems. The guidelines act as solutions or mitigation strategies for the problems. An observation describes the results of an

application of a guideline. Besides this, many different kinds of experience like artifacts developed during projects are to be stored in the EB. Each is called an experience package. In addition, these experience packages are highly interrelated. For example, projects produce deliverables in the form of slide presentations and reports. Slide presentations may be summaries of reports. Observations and problems are gained during a project while a particular business process was performed, that is, we have to deal with context-sensitive experience. Such kind of experience is unique in the sense that the same context will not recur. Therefore, people will be searching for experience that has been gained in *similar* contexts. Both, the requirement for supporting different kinds of interrelated experience packages and the need for context-sensitive, similarity-based retrieval, demand a specialized technical infrastructure for the EB.

These are common requirements for an EB [Tau00]. Our solution to meet these requirements is INTERESTS (Intelligent Retrieval and Storage System) [AB+00]. It consists of a general purpose browser for accessing and presenting the EB contents using a standard web browser, an EB server synchronizing (and logging) access to the EB, and a commercial CBR tool (CBR-Works/orenge from tec:inno, Germany; e.g. [Sch99]), which is used for the actual EB. Each experience package is implemented as a "case" based on a structural CBR approach [BB+99]. This includes a domain ontology for modeling the different types of case concepts, formal and informal case attributes together with the respective similarity measures, as well as relations between cases.

Within an experiment the benefits of this EB approach have already been demonstrated [Tau00]. Until now we have gathered nearly two years of operational experience in maintaining COIN, and we have successfully adapted COIN to partners/customers. Based on this experience we have widened the requirements of COIN towards an organization-wide information and knowledge management system. Other applications not yet considered (e.g. human resource and educational systems) may deliver valuable information, too. Additional information can lead to a more precise and better aggregation and adaptation of knowledge to users needs, but also requires the integration of the respective applications.

4 Experience Base Buildup Method

In [Tau00] a method for the development of EF/EB[1] is presented. The method is called DISER, a methodology for designing and implementing software engineering repositories. DISER accelerates the design of an effective and efficient management system for software engineering experience. Using the DISER method, the systematic buildup of an experience base is performed in six main steps (see Figure 2).

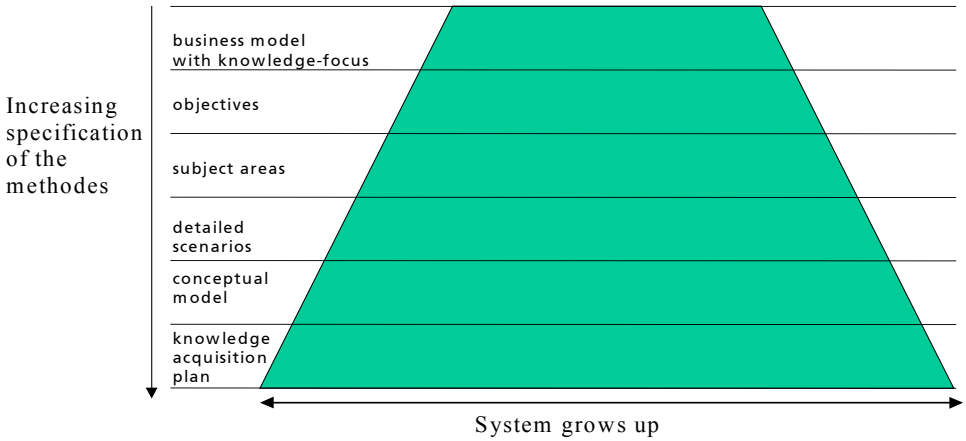

Figure 2 The six main steps of an EB development

A vision of a business model (with focus on the aspect of the usage and production of knowledge and experiences) represents the starting point. This model shows, in particular, where the EF/EB can support the experience transfer. Based on this vision concrete objectives are defined, which are to be achieved by the EB. For the definition of the objectives, the interests of the stakeholders have to be considered. For each of these objectives, appropriate success criteria are defined, which allow measuring the progress concerning the objectives. Based on the business model and objectives, relevant subject areas are identified and selected, which are expected to contribute to the achievement of the objectives. As soon as objectives and relevant subject areas are known, acquisition and usage of experiences can be described with scenarios. In the context of these scenarios, the need for information is identified in more detail. This allows developing a conceptual model for the experiences. In

[1] The buildup of an EB always requires a surrounding EF either through (re-)using an existing one (i.e. the EB is extended) or through infusing one in the respective organization. While DISER considers both, in this paper we focus on building up an EB only.

the last step, the knowledge acquisition is planned in detail and described in a knowledge acquisition plan.

A development process using DISER creates a schematic top-down documentation for the implementation concept (i.e., conceptual model and knowledge acquisition plan). We call such a documentation *rationale*. Such a *rationale* documents the relationships of the components of the conceptual model and knowledge acquisition plan via scenarios and relevant subject areas to objectives and business model and makes these relationships traceable. A comprehensive documentation of these aspects is also the basis for reuse of this information for the development of similar EBs.

5 The Experience Management Content Framework of the EB

The DISER method demands to set up a vision for the resulting knowledge management infrastructure. Figure 3 presents the Experience Management Content Framework (EMCF). EMCF acts as a vision for a comprehensive management of experience within an organization, thus, representing a generic blueprint of an EB.

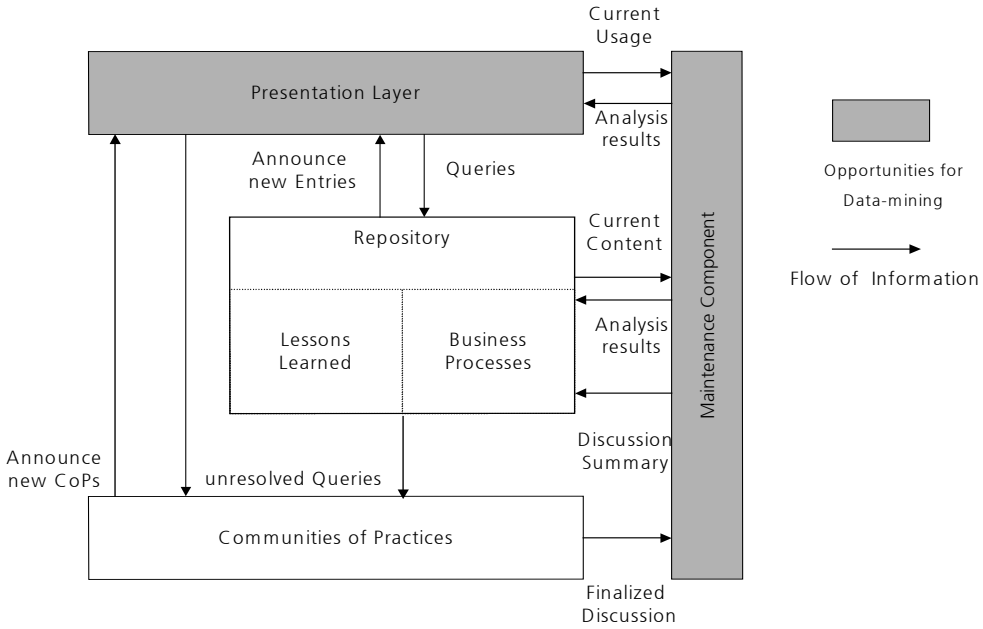

Figure 3 The Experience Management Content Framework (EMCF)

The EMCF consists of four basic components: Presentation Layer, Repository, Communities of Practices and Maintenance Component. The *Presentation Layer* is the interface of the EB to the regular user. It provides (a) uniform access to the information residing within the EB, (b) stores the user preferences and settings, and (c) adapts and aggregates information within the EB based on those preferences (Sec. 8). The *Repository* contains the explicitly captured and consolidated experience of an organization. As mentioned earlier (Sec. 3) for COIN, a combination of business process descriptions and lessons learned was chosen as a starting point (Sec. 6). Further experience management activities can be set up on that base [DJ01]. The *Communities of Practice* component is a forum for the member of an organization for discussion of current problems, questions and open issues (Sec 8). Finally, the *Maintenance Component* supports the EF team in maintaining and developing the content of the EB

(Sec. 7) (i.e., the data within the repository) and the services offered to the organization (via the Presentation Layer). This component offers the second place of a sensible application of data-mining methods: The content as well as the usage of the services can be analyzed to trigger, guide or otherwise support maintenance activities (Sec. 9).

The EMCF components and the communication between them are detailed in the following sections.

6 Managing Business Processes and Lessons Learned as Experiences

Business processes and lessons learned on project management are the two major areas, we are focused on within our EF (i.e., COIN).

6.1 Business Process Descriptions

High quality procedural knowledge, that is, "good" business processes, is a competitive advantage of an organization, since it coordinates its members in an effective and efficient way. Therefore, an area within COIN was to build up and handle process descriptions. It is called IESE Quality Management System, or IQ for short. IQs' objectives can be characterized according to four criteria: (1) the purpose of process models, (2) the origin and (3) usage of the process models, and (4) the modeling techniques. In summary, IQ uses structured text describing empirical and theoretical process models to be executed by human agents. This is detailed in the following.

For the *general purpose of process models*, [CKO92] identifies five different categories: Facilitate human understanding and communication, support process improvement, support process management, automate process guidance, and automate execution. According to this classification scheme, IQ fits into the first category of facilitating human understanding and communication: processes, based on the process descriptions, are executed by human agents (i.e., IESE members). Supporting and enforcing process execution beyond this human-based approach (e.g., by workflow modeling and enactment as in [MH99]) can be regarded as suitable if it does not contradict the creative nature of the business processes. Additionally, two questions have to be asked: (a) Is automation of the processes supposed to leverage a high cost/benefit and (b) is tracking of process status possible by asking the responsible process executor. The experience made with the Electronic Process Guide (EPG) [BV99] showed that web-based process descriptions are a feasible way of distributing process knowledge within creative environments such as software business. In particular, changes to web-based process models can be communicated much quicker than paper-based process models, thus enabling a quick integration of experience.

The *origin of process models* can be empirical (i.e., based on actually executed processes) or theoretical (i.e., reflecting a planned process execution). Process models in IQ have both origins: Some of the process models reflect well established processes (like, e.g., the administrative project set-up), others represent

new procedures (e.g., the reflection of recent changes in the organizational structure of IESE).

The *usage of process* models can be descriptive (i.e., a description of a process) or prescriptive (i.e., intended to be used as an instruction for process execution). The process models within IQ are prescriptive with different degrees of obligation. In general, administrative procedures (e.g., project accounting) have to be followed without exception; best-practice process models like project management procedure are to be seen as recommendations.

The *process modeling technique* of IQ is structured text, which is due to several reasons: zero effort training, straightforward modeling and perpetuation in industrial strength applications. *Zero effort* has to be spent on training, since any IESE member can read structured text without previous training. Furthermore, *straightforward modeling* means that any IESE members can model processes using structured text, if supported by guidelines and the COIN team. This aspect is additionally fortified by the experience in scientific publishing of most of the IESE members. For the modeling itself, any word processing software can be used. Finally, structured text has proven its *industrial strength* in process models of quality management systems [Dil95]. Because of structured text cannot be checked automatically for consistency, all process models are reviewed thoroughly by peers and the COIN team. To facilitate this task, one potential development of IQ could be to enhance the process descriptions by integrating a formal modeling technique that would allow sophisticated automatic consistency checks.

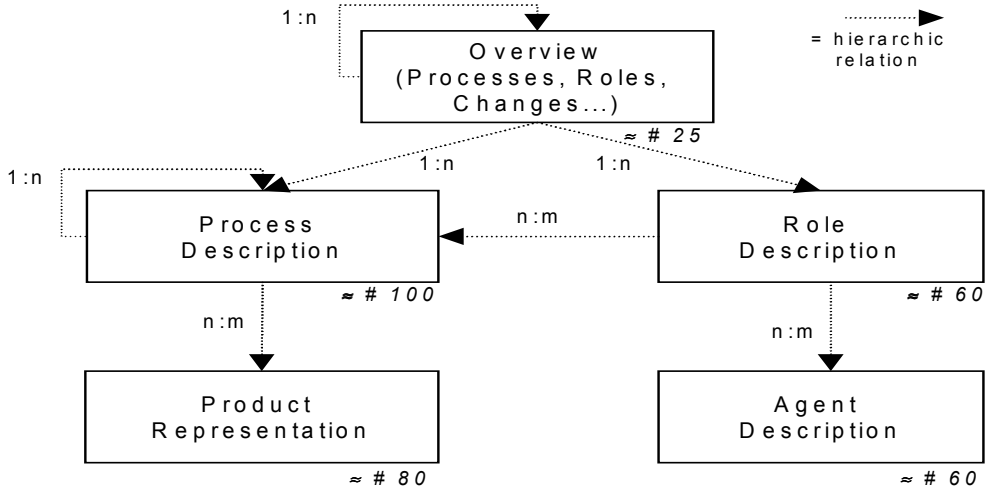

Figure 4 Simplified structure of objects in IQ

The correlation of information objects captured within IQ is shown in Figure 4. Each of these information objects can be linked to other objects:

- *Process Descriptions:* Process descriptions describe the activities captured within COIN (e.g., project management). Complex processes are structured into a hierarchy of super- and sub-processes.
- *Role Descriptions:* Role descriptions describe the roles that are involved in the execution of processes.
- *Agent Descriptions*: Agent descriptions are used within role descriptions to name roles that are performed by a specific IESE member.
- *Product Representation*: Product representation represents documents to be used during process execution.
- *Overviews*: Overviews structure the other objects within IQ to facilitate browsing.

6.2 Capturing and Presenting Lessons Learned

Lessons Learned (LL) at this time are mainly acquired using structured interviews (a) regularly for running projects and (b) at project touch-down meetings. This is exemplary shown in Figure 5. Supported by a questionnaire an experience engineer (COIN-team member) interviews the project team about their experiences, which are collected and summarized in a project analysis report. The report has to be reviewed by the project team. Finally, the project analysis report is split by the experience engineer (EE) into reusable experiences, which are then stored in the EB.

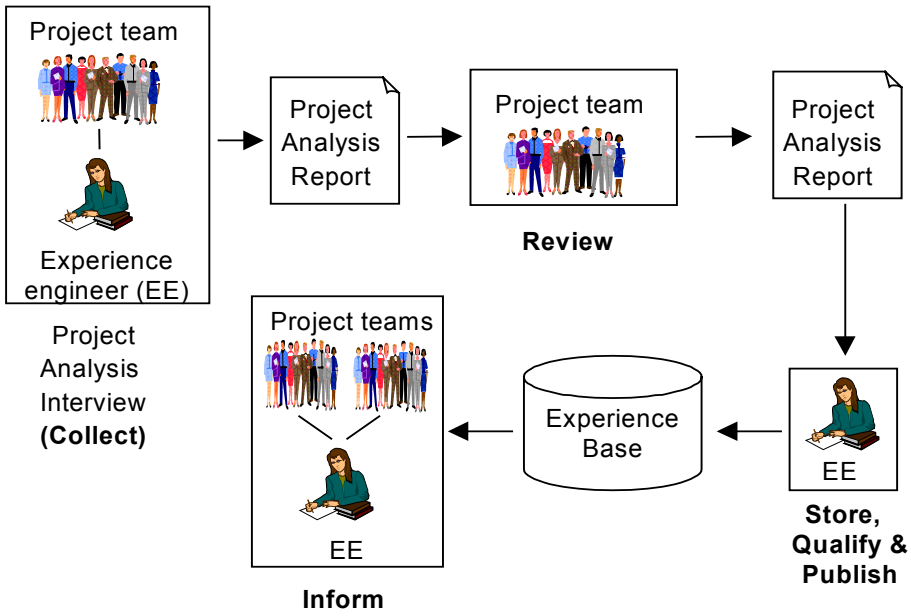

Figure 5 Experience acquisition with project analysis interviews

LL can cover different topics and take on different forms [BT98]. Within COIN, LL about project management are captured. One LL can take on the form of an *observation*, a *problem*, *guideline*, *pragmatic solution*, or an *improvement suggestion*. Each LL is personalized to allow a querying IESE member to ask a colleague for further information. The context of these LL are modeled by the two concepts "project" and "process": *"project"* is a characterization of the project in which the lesson learned was gained (e.g., person month, duration). *"Process"* names the business process and thus the project phase in which the LL was gained. Therefore, a project team member can specify his current environment as well as the current situation to search the EB for similar experiences. Figure 1 shows the interrelations between the context and the different types of LL.

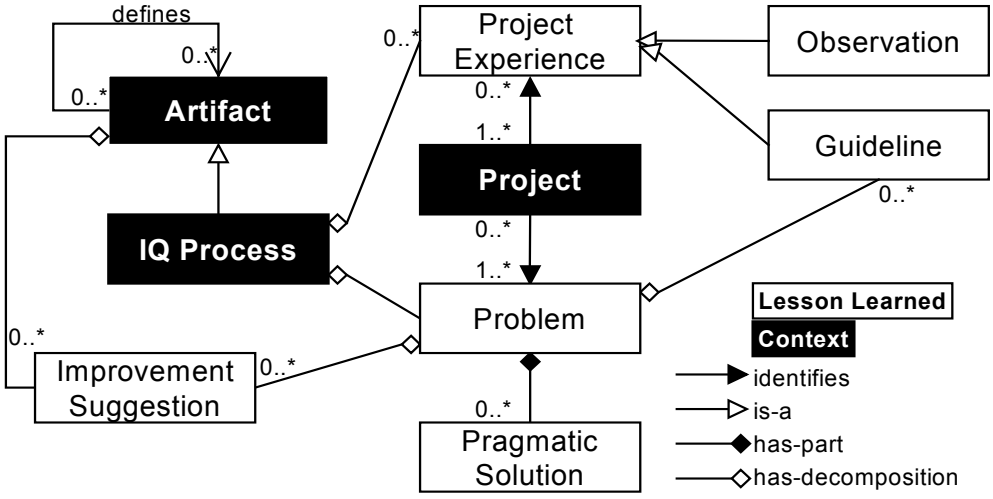

Figure 6 COIN-EF Ontology according to [Tau00]

Observations are facts that are of interest to future projects, often expressing some baseline (e.g., "it took 10% of the total effort to manage the project") or some positive effect (e.g., "the customer was happy because we provided him with a ready-to-use tutorial"). *Problems* are descriptions of negative situations that occurred during a project (e.g., "the expectations of the customer were not met"). Guidelines, improvement suggestions, and pragmatic solutions relate to one or more problems. *Guidelines* are recommendations on how a particular business process should be performed. For example, a guideline could be the following: "Interact with the customer frequently, at least twice a month." An *improvement suggestion* is a proposal to change an artifact to avoid problems that occurred during its usage. *Pragmatic solutions* are sequences of immediate countermeasures taken by a project team in response to a recognized problem. While a guideline aims at preventing a problem from occurring in the first place, a pragmatic solution is applied after a problem has already occurred.

These project management LL (a) complement process execution differently and (b) are integrated into the process descriptions in several ways. Two examples should illustrate this relation: Observations can be used to build mental models or validate assumptions about project work (e.g., customer preferences). Consolidated guidelines can be integrated into process descriptions.

7 Maintenance

The value of a corporate information system tends to degrade with time, caused by external impacts on the organization's environment or by changes within an organization (e.g., the development of a new product). This is particularly true if case-specific knowledge (experience) is stored in the information system, as is typically done in CBR systems, EB systems, LL systems, or best practice databases, because such knowledge is gained almost continuously in daily work [AW00].

Maintenance is of particular importance, for EM, because a (more or less) continuous stream of experience has to be processed [BB+99, NAT01]. For instance, since 1999 our small in-house EB for our 60 researchers has grown annually by 550 lessons learned and we expect an annual growth of 500 LL for the next years. In addition, our EB also includes best practice descriptions on business processes and information on projects as well as the links among these different knowledge and information types. Because the EB should be maintained with low effort and the EF staff as maintenance team can work only part-time for the EF/EB, tool support is highly regarded [NAT01].

High quality of the retrieved knowledge is a main requirement from the system users point of view. The respective quality criteria should be related to organizational goals [Tau00, NF00].

All this demonstrates that maintenance has a certain complexity for such systems and is a knowledge-intensive task. Thus, guidance and decision support for maintenance is almost essential to successfully maintain and improve such a system. Due to the variety and number of the knowledge in an EB system, authoring support has to combine human- and computer-based maintenance activities. However, the maintenance knowledge for decision support and specific maintenance tasks is rather acquired "by chance" during continuous operation (so far). Thus, it might take long to learn the required maintenance knowledge for decision support. The problem is that existing methods such as INRECA [BB+99] or DISER [Tau00] only fill the "standard" knowledge containers of CBR/EB systems.

Based on DISER, we are developing - in the context of maintenance - the EMSIG framework, presented in [NAT01], for authoring support for EB/CBR systems, which combines human- and computer-based maintenance activities and respective decision support. EB and authoring support tools are implemented in an integrated system using CBR technology. The EMSIG framework includes an integrated technical solution that operationalizes the support for

EB maintenance regarding cases and conceptual model using specific maintenance knowledge.

7.1 Overview on EB/CBR Maintenance Knowledge Types.

Quality knowledge describes how the quality of the EB/CBR system is measured and the current status of the system with respect to quality as well as the rationale for the definition of quality [Men98]. Quality knowledge deals with quality aspects of the EB/CBR system as a whole, that is, the EB's contents and conceptual model as well as retrieval mechanisms, usability of the user interface, etc. An example for content-related quality knowledge is a definition of measures for the utility or value of single cases [NF00]. There are several types of quality knowledge that are related as follows: The measures define what data is collected. The data collection is performed automatically or manually by respective data collection procedures. The collected data is analyzed using predefined models or procedures. The results of the analyses can be used for justifying an EB and as input for decisions about maintenance [NAT01, NF00].

Maintenance process and procedure knowledge defines how the actual maintenance activities are performed. The actual maintenance can be performed as a mix of automatically, tool supported and manually performed activities *(maintenance procedure*s). To improve guidance for the maintainers, descriptions of these processes are provided (e.g., detailed description of the acquisition of new cases through collecting cases, reviewing these cases, and publishing them in the case base - see DISER [Tau00] and INRECA methodology [BB+99] for examples). To combine manual and automatic maintenance, a maintenance process can have automated subprocesses/steps, which use input from or provide input for manually performed steps.

Maintenance Decision Knowledge links the quality knowledge with the maintenance process knowledge. It describes under what circumstances maintenance processes/procedures should be executed or checked for execution. Such maintenance knowledge can be described in an informal manner as *maintenance policies* [LW98], which define when, why, and how maintenance is performed for an EB/CBR system. The "why" addresses not only the reason of maintenance but also the expected benefits of the maintenance operation, which should be related to the objectives of the EB/CBR system or to the general goal of maintenance (i.e., to preserve and improve the EB's value [NAT01]). Since these objectives are typically very high-level, it is not very meaningful to address the EB objectives directly. Instead, we use a refinement of the objectives: the quality criteria from the evaluation program or the recording methods. The "how" is a combination of maintenance processes and procedures with additional steps as "glue."

7.2 Acquiring Maintenance Decision Knowledge

The principle of acquiring and developing maintenance decision knowledge is to derive operational maintenance knowledge from three major sources: (1) a knowledge life-cycle model, (2) artifacts developed and information gained during buildup, and (3) the parts of the quality knowledge defining what, when, and how to measure. From these sources, we derive rather informal maintenance policies. Using our method we developed maintenance decision knowledge for our in-house EB as well as for an industrial EB project in the telecommunications domain. In addition, generic, well-tested maintenance policies from CBR research and practice can be reused for general aspects (see [LW98] for an overview and [LS+01] for the state of the art). Before using these, their application constraints must be checked and analyzed carefully. In the following, we focus on the knowledge life-cycle model as a major source and on how to derive maintenance policies from it. Detailed information on the other sources and on how to derive maintenance policies from the sources can be found in [NA01].

Trigger: validity ratio < X% and
number of related observations "phrasing not comprehensible or misunderstood" > N
Actions:
(a) The case has to be rephrased and tested regarding its understandability. The observations
that are considered during rephrasing are deleted. The latter implies an increase of the validity
ratio of the case.
(b) The quality criteria for acquiring, reviewing, and storing the cases have to be checked regarding
their effectiveness for ensuring the comprehensibility of the stored cases.
Expected benefits: The description is easier to comprehend.

Figure 7 Example of a maintenance policy derived from a knowledge life-cycle model.

A *knowledge/experience life-cycle model* describes the basic idea of how to maintain and improve knowledge and experience over time. Thus, it is the basis of any further refinement of the maintenance process, in particular, with respect to the case container. The life-cycle model addresses two major issues: (1) the life-cycle of cases of one type (e.g., [MH00]) and (2) the life-cycle of knowledge over different types of cases. The life-cycle of cases of one type mainly addresses validity issues and/or its revision status. *Validity* describes how general an experience is and how much one can trust the experience to be successfully reused in its anticipated application context. To integrate validity issues into the life-cycle model, an operational definition of validity is required. This definition can be in a qualitative [Mue99] or quantitative manner (e.g., ratio of successful reuses over overall number of reuse attempts [NA01]). The life-cycle of knowledge over different types of knowledge/cases describes when and how cases are transformed from one type of knowledge into an-

other. E.g., a group of related LL can be transformed into a best practice description that is more comprehensive, mature, and for a broader audience [ABT98].

The objective of *deriving maintenance policies* is to describe –in an informal manner– when, why, and how to do maintenance on an EB system. There are two major issues in the knowledge life-cycle model that are refined using maintenance policies:

1. The ways of dealing with failed reuse attempts have to be defined: This is defined by a combination of monitoring the quality/validity of the knowledge in the EB and proposing respective actions. These maintenance policies mainly refer to corrective actions and fix problems that were encountered during reuse of a case. For example, a case was misunderstood and applied incorrectly several times, which requires two actions: rephrasing and checking if the acquiring, reviewing, and storing of cases is unreliable or inaccurate (see Figure 7).
2. The transformation of experience of one type into another is performed under specific conditions and for certain reasons. These conditions and reasons have to be identified and related actions are outlined. Together reason and related action form a maintenance policy.

7.3 Tool Support for Maintenance Decision Making

To allow tool support, the maintenance policies are further formalized as *maintenance guideline*s. The maintenance guidelines have the same basic structure as the maintenance policies. To allow tool support, the following major changes and extensions are made with respect to maintenance policies (see [NAT01] for details):

A partial or complete formalization of the "trigger" is required to allow an automatic tool-based checking of the trigger. The formalized parts of the trigger can refer to items from the standard containers as well as measures, user feedback, periodic events, or events such as the end of a project. For more advanced analyzes, the trigger can also refer to results from analysis tools such as data or text mining tools (see also Sec.9). The parts of the trigger that cannot be formalized are included in the maintenance guideline for manual checks by the responsible role. In case the actual trigger cannot be formalized at all, the respective guideline can be triggered periodically as a reminder, and the actual condition check is done manually by an EF staff member (COIN team).

The "actions" refer to human-based maintenance processes, automatic maintenance procedures, and the "glue" among these. Thus, it is possible to combine automatic procedures with human-based execution of processes. Looseness or tightness of this integration depends on the tools, for example, one

could export cases for analysis with a data mining tool each month (loose integration) or a data mining component could be run automatically (tight integration).

Using a CBR tool that follows the structured approach [BB+99] and supports different types of cases (e.g., Orenge from tecinno/empolis GmbH, Germany), the maintenance guidelines idea can be integrated into the existing CBR system (e.g., using EMSIG's maintenance decision support components [NAT01]): The structure/domain model is extended respectively and the actual maintenance guidelines are stored as cases in the system. An active component is required to analyze the trigger conditions of the maintenance guidelines and to perform the respective actions, i.e., launch the respective maintenance tool/component for the action or notify maintainers about the maintenance processes to be performed for certain cases, vocabulary, etc.

8 New Strategies for Capturing, Process, Disseminate and Exchange Knowledge

Knowledge is actually identified as "fourth factor of production"[2]. Therefore, unstructured, not personalized flooding with information can be counterproductive for building up and exchanging knowledge. For an improved support of our employees we are (a) moving from a "pull" to a "push" strategy in the sense of providing the right information at the right time, (b) developing more flexible and faster mechanisms for sharing information, and (c) developing a concept for aggregation and adaptation of information to users' context and needs.

8.1 "Push" of Information/Knowledge

We do not want to burden users with overhead for searching information or asking for experience. Our solution grants with a single point of access, admission to all knowledge and information produced in an organization, only restricted by access rights defined by (a) the organization in form of the employee's role within it, (b) the projects and the according role the employee plays, and (c) the owner of a piece of information. Therefore, a user interface has to be developed according to the presentation layer of Figure 3.

With his login in combination with stored user data (his organizational role, project roles, and skills) and a chosen view (concrete project) the user provides the actual context, for example: "role: developer; project: x; task: code testing" (the task is determined from the project plan). The given context is used to deliver knowledge collected within similar contexts without an explicit user query ("push" of information). The user can ignore this but, hopefully, he will at least evaluate the utility of the delivered information within his actual context. The evaluation is used to "educate" or "edge" agents for users' business and personal information needs. Personal needs can be context-sensitive or free of users choice.

In the case of new, improved, or changed content the user will be informed automatically if he has registered for this service. This can happen by mail or directly after his login on the physical presentation layer. The user has registered only once for those components he wants to be informed of.

Additionally, he can send specific queries to the EB ("pull") (see Figure 8).

[2] Besides work, capital, raw material

8.2 Community of Practice Base (CoP)

To get information, current users have to send a query to the EB. As practice shows, sometimes there is no appropriate case experience package available for the specified problem. The user has to find his own solution, which tends to be available only to a very small group of people, unless he tells the COIN team about the gained experience. According to [NT95] this means the externalization of implicit knowledge towards explicit knowledge. Currently project experiences are collected periodically and at the end of a project using project analysis interviews (i.e., a structured interview for acquiring lessons learned from project members). The project members tell their experiences within those interviews to the EF team, whose members are responsible to extract and derive lessons learned in form of guidelines, observations, and problems and to put them into the EB (Sec. 6.2). For some problems, occurring within the projects this process is too slow. To present a solution, we are aiming at extending the EF through a more flexible concept, namely communities of practice (CoP). They can be treated as tool support for task-oriented collaborative learning pointing out team-learning and collective intelligence [Klu99].

CoP handle specific problems for which there is no information in the EB available, so far. In such a case the query is, with the agreement of the user, forwarded to the project-specific community of practice (PCoP) (see Figure 8). Every project member who currently has got a view on this project, will see the question nearly at the same time, this can also include customers or partners outside the organization. They can assist by providing their own experience and simultaneously, they extend the knowledge base. Intuitively, the CoP supports the collection of tacit, personal knowledge. If after a while (the asking user can give a deadline) nobody answered the question (sufficiently), it is sent to the organization-wide CoP (if the user agrees), where every user can answer the question. In addition, it could become one of the duties of some very experienced IESE members to look up the CoP at least once a week. The asking user should be able to evaluate the utility of the given answer according to his specific context by giving bonus points. These points can be gathered and an award like the "Expert of the Month/Year" can be instantiated, which is expected to motivate people to use this feature of COIN. (For a knowledge market a new currency for knowledge units (KU), similar to a stock market (supply and demand) can be regarded).

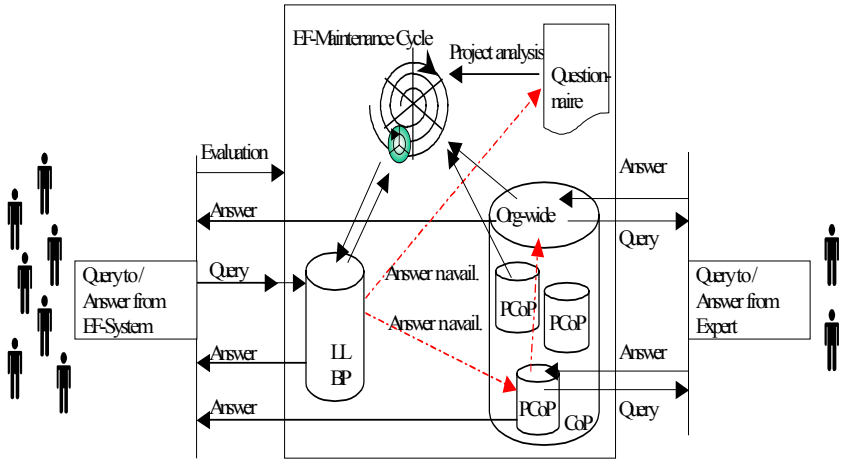

Figure 8 Correlation: Experience base (EB) and communities of practice (CoP)

To support project analyses in a more specific way, such questions can also extend the questionnaire for the interview. The project member who sends the query now should be able to answer the question, because of experiences, that solved the task, told by others through the CoP or made by himself. In this context it seems to be important to mention that the collection of both positive *and* negative experiences is necessary in case of a knowledge network [BJA01]. An approach to archive project-specific CoPs within the project will be developed to avoid loss of experience.

Another part of the operative work of an EF besides the collection of experiences during project analysis is the maintenance of the EB content. With every new input to the EB, existing experience packages can be confirmed or questioned (Sec. 6). This work will be supported by the introduction of the utility evaluation by the users. The EF Maintenance cycle (Figure 8; see also Sec. 7) shall symbolize the necessary activities. Rejected content can be discussed and widely evaluated using the CoP. At any rate, questions not yet answered or rejected content have to be considered as hints for maintenance, that is, in-depth analyses. The results will help to improve (a) EB content but also (b) information aggregation and adaptation, which includes education of the agents through the user by "carrot (bonus points) and the stick (rejection)". To save expenses, maintenance of the CoPs is done mainly automatic by using the given bonus points and timestamps. If the information is regarded as worthless repeatedly and the timestamp is reached it is moved to trash. In contrast, information regarded as valuable are forwarded to the EF-Maintenance-Cycle for a final inspection before it is stored in the EB.

8.3 Aggregation and Adaptation of Information

Every member of an organization or, more abstract, every role has different needs with regard to the granularity of information. Stepping higher on the organizational or project level, information has to be aggregated and adapted more and more with respect to the urgency and criticality. This concept is well known in data mining methodologies. Extending these approaches we are dealing with experience in form of un-/structured documents. Data mining shall support the gaining of valuable information to confirm/reject experience.

The user gets standard information in addition with an attribute, telling him about the degree of utility (personalized or evaluated experience) and the name of the author. Highly aggregated and adapted information mostly cannot be assigned to one input source. The level of aggregation and adaptation is then told to the user, so he his able to comprehend the outcome. Only if a state is detected as critical, detailed information (source information) is available, on demand.

While project members need specific and in-depth information about their status within the project, the project leader is more interested in an overview of all project activities. For him it is valuable information that a deviation will occur because of illness of a project member. Experiences available dealing with those cases, regarding the risk plan, can assist him in evaluating the critical potential of this state. If he detects a business-critical state, the information is forwarded on a "red-phone" channel to the respective persons.

Further research work has to be done here.

9 Data Mining in Experience Bases

In our effort to optimize and improve an EF like COIN we are currently evaluating the basic techniques of Data Mining [ES00]. This includes clustering, classification, association rules, and generalization.

Clustering makes it possible to find similar sets of data in large multi-dimensional databases. It can be used to detect unexpected clusters in groups of experiences, how close those experiences are related or what types of deviations from clusters exists. *Classification* is a technique to categorize new experiences into an existing hierarchy of classes. As a side effect to the construction of the classification process it collects data about the borders between classes of experiences. The third technique — *association rules* — is used to find patterns in transaction data. These rules are represented in the form "if ... then ..." and are typically used to predict customer buying behavior. Last but not least *generalization* is used to generate summaries of numerical data or to apply other data mining techniques on more abstract descriptions of the data.

Another important aspect in an experience base with almost exclusive textual data is *Text Mining*. With techniques from this field we can automatically extract "hard" data from experiences and pre-process it for any of the Data Mining techniques. This "hard" data consists mostly of extracted words from the experience itself and can be used to create values for attributes of a classification.

All these techniques can be used to analyze, generalize, and process the experiences within COIN, as well as to support the construction and usage of an EB. To improve COIN we are in the process to develop (a) concepts to discover new knowledge in experiences, (b) a method to support the construction and evolution of EB's, (c) concepts to support COIN users, and (d) methods to detect and improve the "quality of knowledge". Beside these application areas, data mining can be used to support the other presented strategies like "aggregation & adaptation" or "community of practice (CoP)".

9.1 Knowledge Discovery in Experience Bases / COIN

The discovery of hidden and previously unknown knowledge from existing data is the primary goal of data mining. We want to pursue the same goal for the experiences stored in COIN like process descriptions, lessons learned or best practices.

With classification techniques (e.g. decision trees or support-vector-machines) we can pinpoint the borders and decision rules of classes. Clustering helps us to find the center of a cluster that represents "normal" experiences as well as to detect deviations from the core cluster — e.g. either extremely good or bad practices. Through association rules it is possible to supply the user not only with exact or similar experiences he searched for but also provide him experiences other users in similar situations have retrieved. Also it helps us to find trends over time in the experiences or to predict the development of a project based on old experiences [Wie01]. From data about the extraction of experiences by the same or different users of the EB we can find rules about external processes (e.g., one week after extracting a code-component by a coder a tester searches for a test-environment). Likewise information about the time between the retrieval and integration of experiences helps us to remember users to integrate their experiences into COIN.

9.2 Experience Base Construction and Usage Support

Before using an EB it has to be built up. In this process techniques like clustering can be used to find hierarchically dependent classes to build an ontology of the existing experiences. Using the targeted retrieval scenarios (goals) of the EB we should be able to kick-start an EB faster and with a less artificial ontology. Of course, before classifying experiences we first have to find meaningful attributes and values to describe the textual experiences. This is partially possible with techniques from Text Mining like latent semantic analysis [BB99] and decision trees [Per01]. Another application of a data mining technique is the classification of new experiences into COIN based on the currently valid classification.

Besides supporting the construction of an EB data mining can be used to analyze its usage. With clickstream analysis we can infer the retrieval behavior of COIN users to minimize retrieval sessions through a redesign of the classification. User searching behavior and clustering helps us both to find groups of people with similar interests and to "push" new information to the user as soon as it is available and integrated into COIN.

9.3 Quality of knowledge / experience

The quality of the experience is the most important aspect of an EB. Techniques like clustering help us to create more general knowledge by finding similar or exceptional experiences. This artificial and abstract knowledge can be used in any project without looking into many different experiences and adapt them by hand to the current problem/project. With the data mining technique generalization we can generate summaries of projects (e.g., data cubes with summarized status information) or use predefined hierarchies for the classifica-

tion values to reduce the variety in data and find association rules between more abstract data in the higher levels of the hierarchy.

9.4 Support of other strategies

Other EB application areas for data mining can be found as well. With the detection of anomalies in our experiences we can trigger maintenance processes for equalization or updating purposes. By defining or discovering important experiences we can group similar experiences. Therefore, it is possible to aggregate them more easily and transform these experience packages into very adaptable and abstract knowledge. Analogously by classifying new users into user groups they can be immediately integrated into discussion groups (CoP`s) or information-"push"-streams. This makes it possible that a beginner can profit from an EB right from the start.

10 Summary

In this paper we shortly introduced the emerging field of experience management (EM) [Ber01, Tau00]. Among others, the roots of EM lie in experimental software engineering (experience factory), artificial intelligence (case-based reasoning), and knowledge management. EM deals with all kinds of techniques, methods, and tools for knowledge handling as well as the respective knowledge-related processes. The main difference between EM and knowledge management in general is that for EM a (more or less) stream of experience/knowledge has to be processed. We used the Fraunhofer IESE Experience Factory as a means to exemplify some basic EM approaches.

11 References

[AA+95] Althoff, K.-D., Auriol, E., Barletta, R. and Manago, M. (1995a). *A Review of Industrial Case-Based Reasoning Tools*. AI Intelligence, Oxford, UK.

[AB+98] Abecker, A.; Bernardi, A.; Hinkelmann, K..; Kühn, O.; Sintek, M.: Towards a Technology for Organizational Memories, IEEE Intelligent Systems 1998.

[AB+99] Althoff, K.-D., Bomarius, F., Müller, W. & Nick, M. (1999). Using a Case-Based Reasoning for Supporting Continuous Improvement Processes. In: P. Perner (ed.), Proc. German Workshop on Machine Learning, Technical Report, Institute for Image Processing and Applied Informatics, Leipzig, 8 pages.

[AB+00] Althoff, K.-D., Birk, A., Hartkopf, S., Müller, W., Nick, M., Surmann, D. & Tautz, C. (2000). Systematic Population, Utilization, and Maintenance of a Repository for Comprehensive Reuse. In G. Ruhe & F. Bomarius (Eds.), *Learning Software Organizations - Methodology and Applications*, Springer Verlag, Lecture Notes in Computer Science, LNCS 1756, 25-50.

[ABS96] Althoff, K.-D. & Bartsch-Spörl, B. (1996). Decision Support for Case-Based Applications. In: *Wirtschaftsinformatik 1/96*, special issue on case-based decision support (edited by D. Ehrenberg), 8-16.

[ABT98] Althoff, K.-D., Bomarius, F. & Tautz, C. (1998b). Using Case-Based Reasoning Technology to Build Learning Software Organizations. Proc. 13th Biennial European Conference on Artificial Intelligence (ECAI98) Workshop on "Building, maintaining, and using organizational memories" (OM-98).

[ABT00] Althoff, K.-D.; Bomarius, F.; Tautz, C.: Knowledge Management for Building Learning Software Organizations, Information Systems Frontiers 2:3/4, 349-367, 2000; Kluwer Academic Publishers

[ADK98] Abecker, A., Decker, S. & Kühn, O. (1998). Organizational Memory. Informatik Spektrum 21:213-214.

[Aha99] Aha, D.W. 1999. The AAAI-99 KM/CBR Workshop: Summary of Contributions. *Proceedings of the ICCBR '99 Workshops*, II-37–II-44. Technical Report, LSA-99-03E, Department of Computer Science, University of Kaiserslautern: Centre for Learning Systems and Applications.

[AK+89] Althoff, K.-D., Kockskämper, S., Maurer, F., Stadler, M. and Wess, S. (1989). Ein System zur fallbasierten Wissensverarbeitung in technischen Diagnosesituationen. *In: Retti, J. and 111111111Leidlmeier, K. (eds.), 5th Austrian Artificial-Intelligence-Conference*, 65-70, Springer Verlag.

aa[Alt01] Althoff, K.-D. (2001). Case-Based Reasoning. To appear in: S.K. Chang (Ed.), Handbook on Software Engineering and Knowledge Engineering. Vol.1, World Scientific (40 pages).

[AM00] Althoff, K.-D. & Müller, W. (eds.) (2000). *Learning Software Organizations*. Proc. of the 2nd Internat. Workshop (LSO'00), Oulu, Finland.

[AP94] Aamodt, A. and Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches. *AI Communications Vol. 7*, *No. 1*, 39-59.

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

[AW00] Aha, D., and Weber, R., editors. Proceedings of the Workshop on Intelligent Lessons Learned Systems at 17th National Conference on AI (AAAI-00), 2000.

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

[BaS87] Bartsch-Spörl, B. (1987). Ansätze zur Behandlung von fallorientiertem Erfahrungswissen in Expertensystemen. *KI, 4*, 32-36.

[BB99] Berry, M. W.; Browne, M.: "Understanding Search Engines: Mathematical Modeling and Text Retrieval", SIAM Book Series: Software, Environments, and Tools, (June 1999), ISBN: 0-89871-437-0

[BB+99] Bergmann, R., Breen, S., Göker, M., Manago, M., and Wess, S. (1999). *Developing Industrial Case-Based Reasoning Applications - The INRECA-Methodology*. Springer Verlag, LNAI 1612.

[BCC92] Victor R. Basili, Gianluigi Caldiera, and Giovanni Cantone (1992). A reference architecture for the component factory. *ACM Transactions on Software Engineering and Methodology*, 1(1).

[BCR94] Basili, V.R.; Caldiera, G.; Rombach, D.: Experience Factory; In Marciniak, J.J. ed., Encyclopedia of Software Engineering, vol 1, 469–476; John Wiley & Sons; 1994.

[BE+01] Brandt, M., Ehrenberg, D., Althoff, K.-D. & Nick, M. (2001). Ein fallbasierter Ansatz für die computergestützte Nutzung von Erfahrungswissen bei der Projektarbeit. To appear in: H. U. Buhl (Hrsg.), Proc. 5. Internationale Tagung Wirtschaftsinformatik, Augsburg, 19.-21.8.2001.

[Ber99] Bergmann, R. (1999). Preface to the special issue on "Engineering Applications of Case-Based Reasoning. Special Issue of the International Journal "Engineering Applications of Artificial Intelligence", Vol. 12 (6), 661-663, Elsevier.

[Ber01] Bergmann, R. (2001). *Experience management - foundations, development methodology, and internet-based applications*. Postdoctoral thesis, Department of Computer Science, University of Kaiserslautern (submitted).

[BJA01] Bartsch-Spörl, B., Jargon, C. & Althoff, K.-D. (2001). Wissensmanagement in der Praxis - Erfahrungen aus erfolgreichen und weniger erfolgreichen WM-Projekten. Workshop auf der 1. Konferenz Professionelles Wissensmanagement: Erfahrungen und Visionen, Baden-Baden 14.-16. März 2001, (http://demolab.iese.fhg.de:8080/AK-Praktisches-Wissensmanagement/WM2001-WS5/).

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

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

[BR00] Broomé, M. & Runeson, P. (2000). Technical requirements for the implementation of an experience base. In: [RB00], 87-102.

[BSL99] Basili, V.R., Shull, F., and Lanubile, F. (1999). Building Knowledge through Families of Experiments. *IEEE Trans. on Software Engineering 25*, no. 4, 456-473.

[BSL+99] Bartsch-Spörl, B., Lenz, M. & Hübner, A. (1999). Case-Based Reasoning - Survey and Future Directions. In: F. Puppe (ed.), XPS-99*: Knowledge-Based Systems - Survey and Future Directions*. Proc. of the 5th German Biennial Conference on Knowledge-Based Systems, Springer Verlag, 67-89.

[BT98] Birk, A.; Tautz, C.: Knowledge Management of Software Engineering Lessons Learned; Proceedings of the Tenth Conference on Software Engineering and Knowledge Engineering; San Francisco Bay; USA. Skokie, Illinois, USA: Knowledge Systems Institute; 1998.

[Bur98] Burkhard, H.-D. (1998). Extending some concepts of CBR - Foundations of case retrieval nets. In *Lenz et al. (1998a*), 17-50.

[BV99] Becker-Kornstaedt, U.; Verlage, M.: The V-Model Guide: Experience with a Web-based Approach for Process Support; Proceedings of Software Technology and Engineering Practice (STEP); 1999.

[CD00] Conradi, R. & Dingsoyr, T. (2000). In F. Bomarius & M. Oivo (eds.), *Product Focused Software Process Improvement (PROFES'00)*, Springer Verlag LNCS 1840, 391-406.

[CKO92] Curtis, B.; Kellner, M. I., Over, J.: Process Modeling; Communications of the ACE; 1992.

[DA+01] Decker, B.;Althoff, K.-D.; Nick, M.; Tautz, C.: Integrating Business Process Descriptions and Lessons Learned with an Experience Factory; In Professionelles Wissensmanagement - Erfahrungen und Visionen; Aachen 2001; Shaker Verlag S.54-58

[Dil95] Dilg, P.: Praktisches Qualitätsmanagement in der Informationstechnologie; Carl Hanser Verlag, Wien; 1995.

[Din00] Dingsoyr, T. (2000). An Evaluation of Research on Experience Factory. In *[AM00],* 55-66.

[DJ01] Decker, B.; Jedlitschka, A.: The Integrated Corporate Information Network iCoiN: A Comprehensive, Web-based Experience Factory; IESE-Report 031.01/E; Fraunhofer IESE; Kaiserslautern; 2001

[ES00] Ester, Martin; Sander, Jörg: „Knowledge Discovery in Databases: Techniken und Anwendungen", Berlin: Springer-Verlag, 2000, ISBN: 3540673288

[GH80] Gick, M. L. & Holyoak, K. J. (1980). Analogical Problem Solving. *Cognitive Psychology, 12*, 306-355.

[Goo99] Goodall, A. (ed.) (1999). Survey of Knowledge Management Tools - Part I & II, volume 8. *Intelligence in Industry, January & February* 1999.

[GV+01] Gordon, T.F.; Voss, A.; Richter, G.; Märker, O.: *Zeno: Groupeware for Discourses on the Internet;* in Künstliche Intelligenz, Heft 2/01, Seiten 43-45, ISSN 0933-1875, arendtap Verlag Bremen, (2001)

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

[Hen95] Henninger, S. (1995). Developing domain knowledge through the reuse of project experiences. In M. Samadzadeh (ed.), Proc. of the Symposium of Software Reusability (SSR'95), 186-195.

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

[HSW98] Houdek, F., Schneider, K. & Wieser, E. (1998). Establishing experience factories at Daimler-Benz: An experience report. Proc. 20th Internat. Conf. on Software Engineering (ICSE'98).

[JA+01] Jedlitschka, A.; Althoff, K.-D.; Decker, B.; Hartkopf, S.; Nick, M.; Corporate Information Network (COIN): The Fraunhofer IESE Experince Factory. In Proc. 4th International Conference on Case-Based Reasoning (ICCBR'01) Workshop on Process Oriented Knowledge Management

[JH+00] Johansson, C., Hall, P. & Coquard, M. (2000). "Talk to Paula and Peter - They Are Experienced" - The Experience Engine in a Nutshell. In G. Ruhe & F. Bomarius (Eds.), *Learning Software Organizations - Methodology and Applications*, Springer Verlag, Lecture Notes in Computer Science, LNCS 1756, 171-185.

[KM+00] Kalfoglou, Y., Menzies, T., Althoff, K.-D. & Motta, E. (2000). Meta-Knowledge in Systems Engineering: Panacea or Undelivered Promise? *The Knowledge Engineering Review 15(4), December 2000.*

[Klu99] Kluge, A.: Erfahrungsmanagement in lernenden Organisationen;Verl. für angewandte Psychologie; Göttingen 1999;

[Kol93] Kolodner, J.L. (1993). *Case-Based Reasoning*. Morgan Kaufmann Publishers

[LBS+98] Lenz, M., Bartsch-Spörl, B., Burkhard, H.-D. & Wess, S. (eds.) (1998a). Case-based reasoning technology: from foundations to applications. Springer-Verlag, LNAI 1400.

[Leh00] Lehner, F. (2000). Organisational Memory - Konzepte und Systeme für das organisatorische Lernen und das Wissensmanagement. Carl Hanser Verlag

[LS+01] Leake, D. B., Smyth, B., Wilson, D. C., and Yang, Q., editors. Computational Intelligence special is-sue on maintaining CBR systems, 2001. (to appear).

[LW98] Leake, D. B., and Wilson, D. C. Categorizing case-base maintenance: Dimensions and directions. In Smyth, B., and Cunningham, P., editors, Advances in Case-Based Reasoning:Proceedings of the Fourth European Workshop on Case-Based Reasoning, pages 196–207. 1998. Springer-Verlag. [Men98] Menzies, T. Knowledge maintenance: The state of the art. The Knowledge Engineering Review, 73 pages, 1998.

[MH99] Maurer, F.; Holz, H.: Process-Oriented Knowledge Management For Learning Software Organizations, Proceedings of 12th Knowledge Acquisition For Knowledge-Based Systems Workshop 1999 (KAW99); Canada, Banff; 1999.

[MH00] Minor, M., and Hanft, A. Corporate knowledge editing with a life cycle model. In Proceedings of the Eighth German Workshop on Case-Based Reasoning, Laemmerbuckel, Germany, 2000.

[MP90] McGarry, F. & Pajersky, R. (1990). Towards Understanding Software - 15 Years in the SEL. Proc. of the 15th Annual Software Engineering Workshop, NASA Goddard Space Flight Center, Greebelt, MD, Software Engineering Laboratory Series, SEL-90-006, Nov. 1990.

[Mue99] Müller, M. Interestingness during the discovery of knowledge in databases (in German). Künstliche Intelligenz, pages 40–42, Sept. 1999.

[NA01] Nick, M., and Althoff, K.-D. Engineering experience base maintenance knowledge. Technical Report IESE-Report No. 018.01/E, Fraunhofer IESE, 67661 Kaiserslautern, Germany, 2001.

[NAT01] Nick, M.; Althoff, K.-D. & Tautz, C. (2001). Systematic Maintenance for Corporate Experience Repositories. Computational Intelligence 17(2), 364-386

[NF00] Nick, M., and Feldmann, R. Guidelines for evaluation and improvement of reuse and experience re-pository systems through measurement programs. In 3rd European Conference on Software Measurement (FESMA-AEMES 2000), Madrid, Spain, Oct. 2000.

[NT95] Nonaka, I.; Takeuchi, H.: The Knowledge-Creating Company – How Japanese Companies Create the Dynamics of Innovation. New York 1995.

[Per01] P. Perner, Data Mining on Multi Media Data, Chapter 3.1; Decision Trees, Springer Verlag 2001 to appear

[RB00] Ruhe, G. & Bomarius, F. (eds.) (2000*). Learning Software Organizations - Methodology and Applications*. Springer Verlag, LNCS 1756.

[Ris83] Rissland, E. L. (1983). Examples in Legal Reasoning: Legal Hypotheticals. *Proc. IJCAI-93*.

[Rom96] Rombach, H.D. (1996). New institute for applied software engineering research. *Software Process Newsletter No 7*, 12-14, Fall 1996.

[Rom98] Romhardt, K. (1998). Die Organisation aus der Wissensperspektive - Möglichkeiten und Grenzen der Intervention. Wiesbaden: Gabler Verlag.

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

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

[Sch99] Schulz, S. 1999. CBR-Works - a state-of-the-art shell for case-based application building. In *7th German Workshop on Case-Based Reasoning*, Wuerzburg, Germany.

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

[SW98] Stolpmann, M. & Stefan Wess, S. (1998). Optimierung der Kundenbeziehung mit CBR-Systemen. Intelligente Systeme für E-Commerce und Support. Addison Wesley.

[TA97] Tautz, C. & Althoff, K.-D. (1997). Using Case-Based Reasoning for Reusing Software Knowledge. In: D. Leake and E. Plaza (eds.), *Case-Based Reasoning Research and Development*, Second International Conference on Case-Based Reasoning (ICCBR97), Springer Verlag, 156-165.

[TAN00]Tautz, C., Althoff, K.-D. & Nick, M. (2000). A Case-Based Reasoning Approach for Managing Qualitative Experience. In: *[Aha & Weber (2000)].*

[Tau00] Tautz, C.: Customizing Software Engineering Experience Management Systems to Organizational Needs; Ph. D. diss., Dept. of Computer Science, University of Kaiserslautern, Germany; 2000; Stuttgart: Fraunhofer IRB Verlag

[vH+96] van Heijst, G., van der Speck, R., and Kruizinga, E. 1996. Organizing corporate memories. In *Proceedings of the Tenth Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, SRDG Publications, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4.

[Wie01] Wieczorek, Isabella: „Improved Software Cost Estimation — A Robust and Interpretable Modeling Method and a Comprehensive Empirical Investigation", Stuttgart: Fraunhofer IRB Verlag, to appear 2001, Dissertation

Document Information

Title:	Experience Management: The Fraunhofer IESE Experience Factory
Date:	July 2001
Report:	IESE-035.01/E
Status:	Final
Distribution:	Public