A Case-Based Reasoning Approach for Managing Qualitative Experience

Carsten Tautz, Klaus-Dieter Althoff, Markus Nick

Fraunhofer Institute for Experimental Software Engineering (IESE)
Sauerwiesen 6
67661 Kaiserslautern, Germany
{tautz, althoff, nick}@iese.fhg.de

Abstract

The explicit management of project experience (lessons learned on project planning and execution) has become of strategic importance to many organizations. This paper describes how to set-up and run a case-based repository for sharing project experience using an industrial-strength case study. The case study shows that it is both effective and efficient for the information seeker to have an experience repository at his or her disposal. In addition, lessons learned regarding the organization and population of such an experience repository are reported. An outlook describes the potential of additionally planned exploitations of the project experiences.

Introduction

Organization-wide sharing of qualitative experiences (i.e., lessons learned) has been recognized as a means to optimize business performance (Gresse von Wangenheim, von Wangenheim, and Barcia 1998). However, the successful set up of a lessons learned program is still a challenge. At our institute, we have started such a program. It is part of a more ambitious project called COIN (Corporate Information Network) which has the main objective of capturing, distributing, and exploiting the business knowledge of the institute. This paper describes how we proceeded in setting up the lessons learned program and reports on the experiences we gained thereby.

The paper is structured as follows. In the following section, principles and benefits of sharing project experience are discussed. The principles are interpreted as requirements for the structure, which is described in the next section. As the name "COIN" suggests, there are two sides to sharing experience: Utilizing and populating a repository. Hence, a separate section is devoted to the topic of populating. Based on the thus captured experiences, a vision for future exploitation possibilities is sketched. The paper ends with a summary and conclusion section.

Fraunhofer IESE´s Corporate Information Network (COIN)

To understand why the project "COIN" was launched, one has to take a look at the business goals and the situation of our institute: The institute was created to transfer innovative software engineering technologies into practice through applied research. Software engineering technologies are tailored to company-specific needs and are constantly evolving. During the transfer, different people in several departments and groups of the institute gain project experience. People learn what Fraunhofer IESE´s customers need, how the people can transfer and tailor software technologies more effectively and efficiently as well as how to improve the institute's business processes.

Expectations are that, with time, Fraunhofer IESE will transfer more technologies that are increasingly better tailored to companies' needs (using the same effort). To do so, the institute must find better ways to search for, reuse, and tailor technology effectively. Therefore, the sharing of experience is a very immediate and obvious problem. This fact led to the launch of "COIN".

Sharing of Project Experience

The effort required for setting up and running an experience transfer program (especially on the side of the project members for providing experience) forced us to start with a limited scope. In our situation, we basically had the choice between two types of experience to share: (a) experience on the applicability and limitations of software engineering technologies (product experience) or (b) experience on the planning and execution of projects (project experience). We decided to start with the latter, because the former was already covered partly by project reports and scientific publications.

Project experience is always related to a particular business process of an organization (in our case, the planning or execution of a project). In addition, it may be related to a particular situation in a particular project (situation-specific experience), a project in general (project-specific experience), a customer (customer-specific experience), or a topic (technology/competence-specific experience). Alternatively, it may be related to a combination of projects, customers, and/or topics. We refer to anything an experience is related to as the context of the experience. Two types of contexts can be distinguished: The context in which the experience was gained (root context) and the context in which the experience is applicable (application context).

Consequently, it is insufficient to store experience without its context. Without its root context, the application context of the experience cannot be deduced systematically (see below). Without the application context, it is not clear under which circumstances the experience can be taken advantage of.

Over time, the application context can be deduced from the root context(s). A conservative approach is to use the root context as the application context. If the same experience is gained in several different root contexts, a single abstract application context can be constructed.

Experience can also be applied in contexts that are not explicitly documented. However, this decision usually requires a human expert. Since it is not possible to document all aspects of the root context (neither effort-wise nor intellectually since much of it is tacit knowledge), it is important to know a contact person that can provide further information. Hence, experience should never be stored anonymously.

Personalized experience also leads to an increased motivation of utilizing the stored experience. An experiment comparing the effectiveness and efficiency of the traditional human-based experience transfer ("ask your colleagues") and the repository-based experience transfer ("query the repository with personalized experience without asking the originators") clearly demonstrated this (Tautz et al. 2000). The experiential results showed that both approaches complement each other in terms of the project experience found. For example, the effectiveness of the repository-based approach is improved by the human-based approach by at least 50% on average with an error probability of less than 0.1%. Here, the improvement is measured in the number of experience items found by the human-based approach but not by the repository-based approach divided by the number of useful experience items found by the repository-based approach. In addition, the participants of the experiment were asked which of the two approaches they would apply in "real" projects. 28 out of 29 participants answered "both". Many participants added they would use the repository-based approach first to identify interview partners for the human-based approach.

Benefits of Sharing Explicitly Stored Experience

So far we assumed that experience should be stored explicitly (at least partly) to make it available to others. However, we have not discussed the alternative: Set up and use an expert network. For this purpose, we originally conducted the experiment (Tautz et al. 2000).

The results show that the repository-based approach is more efficient than the human-based approach for the seeker of project experience on a statistically significant level of well below 1%. In the experiment, the repository-based approach took 12.5 minutes or less to retrieve one useful experience item in 90% of the cases. In 50% of the cases, it took 3.8 minutes or less. In contrast, the human-based approach took 18.5 and 10 minutes for 90% and 50% of the cases respectively. These numbers are based on the effort needed by the information seeker to get the experience items.

If the availability of the experts is taken into account, the repository-based approach is far more efficient. In the experiment, it took 29.3 hours or less in 90% of the cases and 4.6 hours or less in 50% of the cases to get a useful experience item using the human-based approach. Using the repository-based approach, it took only 12.5 minutes and 3.9 minutes respectively.

In addition, the repository-based approach improves the effectiveness of the human-based approach by at least 50% on average with an error probability of less than 0.5%.

Organizing a Best Practice Repository

Project experience (lessons learned) can take on different forms (Birk and Tautz 1998). The two most important forms are observations and guidelines. 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 a ready-to-use tutorial"). Guidelines are recommendations 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." In contrast to observations, a guideline is always based on some problem that has occurred in the past (e.g., "the expectations of the customer were not met."). In this way, relevant guidelines are restricted: Only guidelines that are really needed (to avoid the recurrence of a problem) are stored.

Another form of a lesson learned is the sequence of immediate countermeasures taken by a project team in response to a recognized problem. We will refer to such a sequence of countermeasures as a *correction*. While a guideline aims at preventing a problem from occurring in the first place, a correction is applied *after* a problem has already occurred.

To store and retrieve the various kinds of qexperience, we use INTERESTS (intelligent retrieval and storage system; Althoff, Tautz, and Bomarius 2000). INTERESTS consists of three major parts (Tautz 2000): Application tools for recording and updating experience over the web, an Experience Base Server for synchronizing access to the experience base, and CBR-Works from tec:inno GmbH, Germany. The latter is a case-based reasoning tool allowing similarity-based, context-sensitive retrieval. Case-based reasoning has been recognized as a suitable technology for implementing knowledge management applications (Gresse von Wangenheim and Tautz 1999, Göker and Roth-Berghofer 1999). The various experience forms are represented as different case concepts. We defined the following concepts: Observation, guideline, problem, correction, business process, and project. The latter two concepts are used to store the root and application context of the experience. Semantic relationships between cases are represented by references. For example, an observation

references the business process for which it is relevant and the project in which it was gained (root context). All cases are represented by attribute value pairs. Values can take on the special value "don't care." This allows us to store *abstract* contexts. For example, a project case with all attribute values set to "don't care" except for the attribute "customer" represents a customer context. Experiences referencing this project case are thus customer-specific. Similarly, topic/competence-specific experience can be represented.

Using this structure, the state-of-the-practice is represented by the descriptions of the business processes, which are complemented by context-specific observations, guidelines, and corrections. The similarity-based querying facility of INTERESTS/CBR-Works allows finding experience that was captured in contexts similar to a context at hand. Thus, potentially applicable experiences are identified (even if the application context has not been generalized from the root context yet). The personalized experience enables a project member to ask the original experience provider for more detailed information if necessary.

Figure 1 shows a (simplified) exemplary guideline with its context. The example indicates that not all experience is stored explicitly. For instance, the fact that Spearmint™ is an innovative process modeling tool developed in-house is not stored because it is known to everybody at the institute. The guideline is applicable whenever a project is set up for projects similar to "CAP" (application context). The root context is given by the problem underlying the guideline. If a project manager (responsible for setting up a project) is in doubt whether to apply the guideline, he or she can look at the underlying problem and decide whether the problem might also occur in the project at hand. Even if the guideline is not applied and the problem recurs (later in the project), the stored correction will probably lead to a quick solution. The project depicted in Figure 1 acts as the root context for the problem. It also further restricts the application context of the guideline (see dotted line). A more abstract project description may be used as the application context for the shown guideline (only "Spearmint™" is relevant for the applicability of the guideline). In fact, for the customer X the guideline needs not to be applied, because the answer is already known. The description of a guideline for customer X should state "do not use Spearmint™ for X".

Populating the Case Base

Experience needs to be explicated and stored before the repository-based approach can be applied. To do so requires the performance of the following steps (Althoff et al. 1999):

- Collect: In a first step, the experience must be collected. In our case, this was done using project analysis interviews. At our institute, such interviews are conducted either at the end of a project (Collier, DeMarco, and Fearey 1996) or — in case the project has a duration of more than nine months — periodically, that is, every six months. The interview results are documented as *project analysis reports* (PARs). A PAR

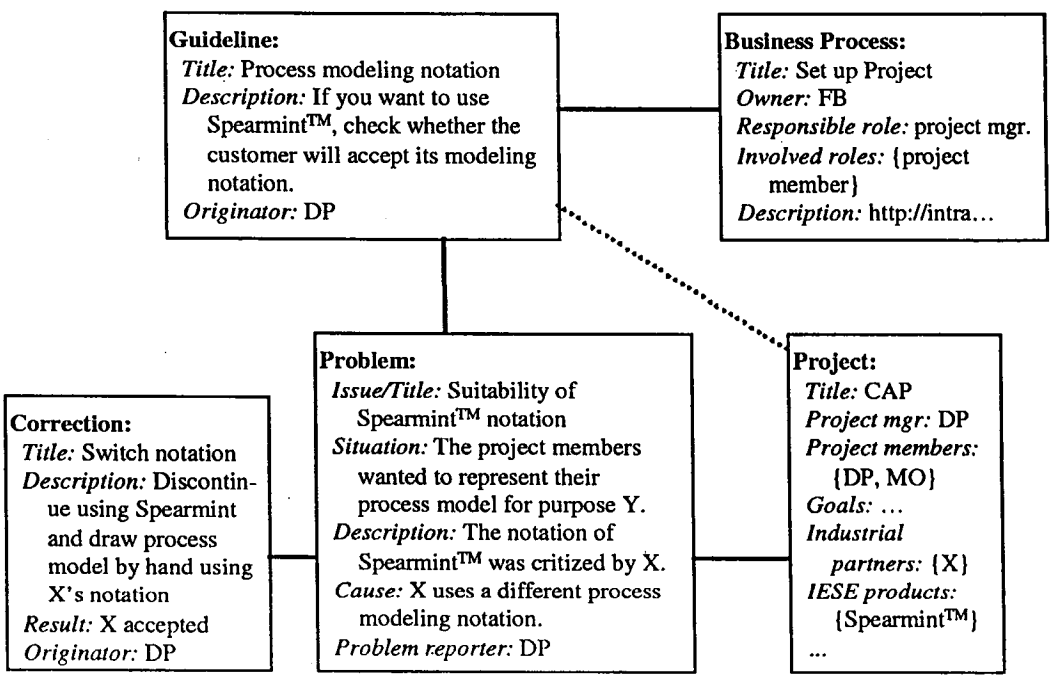

Figure 1: Excerpt of the case base for an exemplary guideline

contains an updated characterization of the project, things to watch out for in similar projects, things that went well, and things that the interviewed project team would do differently if it had to do the same project again.

- Store: In the next step, the collected experience is stored by copying it into the repository, splitting the experience into reusable parts, and initially characterizing each reusable part. In our case, the PARs are split into individual cases: The project characterization in the case base is updated; problems, corrections, guidelines, and observations are extracted out of the PARs and entered into the case base.
- Qualify: Each case is qualified by analyzing its quality (e.g., its comprehensibility) and checking whether a similar case is already stored in the case base. If a similar case is already stored, the new case may be rejected, be merged with the already stored case — possibly generalizing its context —, or replace the stored case.
- Publish: After the new experience has been qualified, it is made available for retrieval, thus enabling the sharing of the new experience.
- Inform: Finally, everybody who may be interested in the new experience is informed.

At our institute, the COIN team is responsible for populating the case base. This includes making the appointment for the interview, conducting the interview, writing the PAR, and all follow-up activities from storing to informing. Such a dedicated team for populating is necessary, because providing experience to future projects is not one of the (prime) objectives of a project (Basili, Caldiera, and Rombach 1994). Therefore, it cannot be expected that project teams provide experience on a voluntary basis. In fact, many of the interviewees said they are willing to participate in future interviews, but would not organize such a meeting on their own.

Although the population seems quite effort-intensive at first, it is important to recognize that only a systematic process guarantees high quality experience. In return, high quality experience encourages its usage, because it is perceived as useful. Thus, experience returned as the result of a query should exhibit a minimal quality.

At Fraunhofer IESE, interviews take at most two hours. 83% of the interviews are one hour long or shorter. The total effort involved on side of the COIN team is approximately four person hours per interview, whereas a project member spends only about 1.7 hours per project and interview including preparation for the interview and proofreading the PAR, This effort is negligible for projects running several months.

The particular process of capturing experience (as described above) also has the positive side effect that most project members view the collection step already as being beneficial (without the utilization of the repository!). During the interviews, the project members learn how their colleagues view(ed) the project and thus have a chance to reflect on the project — something that is rarely done during everyday business. Consequently, they get feedback they would not have gotten otherwise. In addition, they learn what to look out for in future projects. This is reflected by the results of the survey we conducted. In this survey, 34 project members were asked to rate the importance of project analysis interviews. 60% responded with "very important", while 31% responded with "important". Only 2% answered with "unimportant".

Envisioned Usage of Lessons Learned

Up to now, the sharing of project experience has been the focus of COIN. However, the potential of the captured project experiences goes far beyond that. This section outlines future usage scenarios of the captured observations, guidelines, and problems.

Organizational Problem Solving

During the population of the case base, critical or suboptimal project situations are stored as problems in the case base. A project-independent improvement squad meets periodically (e.g., once a month), scans the problem list, and assigns so-called *creativity teams* to solve clusters of similar problems. In this way, solutions that are optimal across the institute can be developed as opposed to the locally optimized corrections developed within the projects. Recurring problems can be avoided by providing guidelines or by changing the description of business processes.

In the context of COIN, we developed an improvement process with the following outstanding characteristics:

- Identification of small problems: Due to the active capturing of problems during the project analyses, a constant supply of problems is guaranteed. Without such an active "problem collection procedure", only serious problems that deeply move project members would be reported. In the past, problems that make project members feel uneasy (but are not serious with respect to project results) were typically not reported.
- Effective problem solving: Due to the systematic problem-solving process, problems are not forgotten (or ignored). Problem reporters are kept informed at every stage of the problem-solving process. This shows employees that their problems are taken seriously and thus encourages problem reporting.
- Efficient problem-solving: The effort to solve a problem depends on the complexity of the problem. A typing error is corrected by a single person, more complex problems are discussed by the improvement squad and solved by an individual person, whereas real complex problems are solved by an assigned creativity team.

This improvement process will be put into place mid 2000.

Consolidation of Project Experience

As already outlined in the beginning of this paper, the context of project experience can and should be

generalized. However, from time to time it is also interesting to take a look at all observations, guidelines, and problems related to a particular business process. Statistics can be created (e.g., what percentage of projects were completed on time within budget). In addition, general trends might be identified (e.g., "what parts of the process description are hard to comprehend?"). Context-specific guidelines for problems, which have occurred often in various contexts, may be "upgraded" to general guidelines and thus become part of the process description.

The performance of such analyses requires a considerable number of stored lessons learned. For our institute, such a cross-project analysis is planned in the second half of the year.

Expanding Project to Process Experience

The management of project experience is of prime importance to our institute (project planning and execution are knowledge-intensive processes). However, the experience management processes described in this paper can (and will) also be applied for business processes other than project planning and execution.

Conclusion

This paper describes how project experience (lessons learned about project planning and execution) can be shared across projects using a case-based reasoning approach. One of the fundamental underlying principles of our approach is that project experience is never stored without its context. Case-based reasoning enables us to retrieve experiences that were captured in a context similar to the one at hand. Experiences on implementing such a lessons learned program at Fraunhofer IESE are reported. On this basis, scenarios for future exploitations of the captured experiences are presented and discussed.

The case study described in this paper must be seen as a first step towards a comprehensive experience management program. Such a comprehensive experience management should not only include the management of process experience (a generalization of project experience to all business processes). Instead, the management (i.e., systematic definition and capturing) of products, competencies, and skills available at the institute should be an integral part of comprehensive experience management. Currently, we are developing an infrastructure to support this kind of experience management.

References

Althoff, K.-D.; Birk, A.; Hartkopf, S.; Müller, W.; Nick, M.; Surmann, D; and Tautz, C. 1999. Managing Software Engineering Experience for Comprehensive Reuse. *Proceedings of the Eleventh Conference on Software Engineering and Knowledge Engineering*, Kaiserslautern, Germany, June 1999. Skokie, Illinois, USA: Knowledge Systems Institute.

Althoff, K.-D.; Bomarius, F.; and Tautz, C. 2000. Knowledge Management for Building Learning Software Organizations. *Information Systems Frontiers*. To appear.

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

Birk, A. and Tautz, C. 1998. 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.

Collier, B.; DeMarco, T.; and Fearey, P. 1996. A Defined Process for Project Postmortem Review. *IEEE Software* 13(4): 65–72.

Göker, M. and Roth-Berghofer, T. 1999. Workshop on "The Integration of Case-Based Reasoning in Business Processes". In: *Proceedings of the ICCBR '99 Workshops.* Technical Report, LSA-99-03E, Department of Computer Science, University of Kaiserslautern, Germany: Centre for Learning Systems and Applications.

Gresse von Wangenheim, C. and Tautz, C. 1999. Workshop on "Practical Case-Based Reasoning Strategies for Building and Maintaining Corporate Memories". In: *Proceedings of the ICCBR '99 Workshops.* Technical Report, LSA-99-03E, Department of Computer Science, University of Kaiserslautern, Germany: Centre for Learning Systems and Applications.

Gresse von Wangenheim, C.; von Wangenheim, A.; and Barcia, R.M. 1998. Case-based reuse of software engineering measurement plans. *Proceedings of the Tenth Conference on Software Engineering and Knowledge Engineering*, San Francisco Bay, USA. Skokie, Illinois, USA: Knowledge Systems Institute.

Tautz, C.; Althoff, K.-D.; Nick, M.; and Ochs, M. 2000. An Empirical Investigation on the Effectiveness and Efficiency of Utilizing Repositories for Qualitative Experience. IESE-Report No. 016.00/E, Fraunhofer IESE, Kaiserslautern, Germany.

Tautz, C. 2000. *Customizing Software Engineering Experience Management Systems to Organizational Needs.* Ph. D. diss., Dept. of Computer Science, University of Kaiserslautern, Germany. To appear.