Organizational Learning via Case Studies in Software Projects

Eike Thaden, Ludger Bischofs and Mathias Uslar
OFFIS, Business Information and Knowledge Management
{eike.thaden|ludger.bischofs|mathias.uslar}@offis.de

Ralf Reussner
University of Oldenburg, Software Engineering Group,
reussner@informatik.uni-oldenburg.de

Abstract: An approach employing case studies for gaining knowledge within software migration projects is presented. First the empirical method case study is defined and later compared to the method of post mortem analysis. We then apply the case study method to the domain of migrating software and discuss the advantages within the context of two industrial case studies conducted with partners. These two case studies are embedded into a larger project investigating knowledge management in software development processes.

1 Introduction

Organizational learning is a powerful tool to improve the performance of organizations. The goal is, that people at all levels, individually and collectively, are continually increasing their capacity to produce results they really care about [Kar01]. The continuous acquisition of knowledge is particularly important for software development organizations, because software development is knowledge-intensive [Rob99]. This means that the underlying processes are non-trivial, comprise many decisions and are not limited to a single knowledge domain. Knowledge relevant for software engineers encompasses both technical knowledge on software technologies and application domain knowledge [Tiw04].

This article focuses and motivates the necessity to acquire the knowledge resulting from projects with teams of software developers. The systematic improvement of software development processes is a crucial part within the knowledge management in software developing organizations.

While individual experience of single project members in such projects may be a source of knowledge which should be shared organization-wide, there exist other types of project knowledge much harder to discover. Each piece of knowledge can be seen as part of a huge puzzle comprising all project knowledge. For the success of projects some pieces of the puzzle might be very important ones, others are unimportant and some even counterproductive. How can we differentiate between those kinds of experience?

The contribution of this paper is to discuss case studies as scientific methods to analyze

certain aspects of software development projects including usefulness of process models for software development with a focus on technology migration. The results can be applied by development organizations to evaluate their projects or by external consultants to evaluate projects of application partners. This paper is organized as follows. First, we discuss the instrument of a case study from a scientific view and introduce the GQM approach and alternatives to case studies. We then present a transfer of the concept to the software migration domain and show problems which can be adressed and measured by the presented approach.

2 Case Studies

One way to evaluate the influence of methods and techniques on successful completion of a project is using case studies. A case study is an empirical study to objectively investigate a new technology in a somewhat realistic setting [Gla04]. The first step in conducting a case study is to formulate one or more hypotheses to examine. For example, such a hypothesis could be *By using knowledge management techniques, the flow of information between project members can be improved.* The next step should be a formalization of the hypotheses making them measurable. The main problem to address by case studies is the relative low level of control over environmental settings in industrial software projects. These settings are determined by budget, time-to-market or similar constraints. Case studies must be designed carefully such that undesirable but not avoidable environmental threats on the research subject can be incorporated in the analysis without making results unreliable.

2.1 Internal vs. External Case Studies

For research facilities making the choice of appropriate projects to study is often a big problem. They have to find industrial partners not too far away with projects suitable for evaluation who are willing to corporate and to entrust them vital information. This kind of case study can be categorized as *external*. On the other hand case studies conducted by organizations to improve themselves can be categorized as *internal*. Choosing projects for internal case studies is significantly easier because only own projects have to be considered as potential study objects. Another advantage is that internal case studies provide higher control over the projects environment (employees, technologies, time). Managers with the necessary power of decision and a research budget can decide to use a promising technology in a suitable project and evaluate the consequences. However, many organizations do not have qualified researchers for internal case studies. Another disadvantage is the lack of independency of the researchers of the organization.

2.2 Goal Question Metric

One popular approach to structure case studies and avoid threats to validity is the Goal Question Metric method originally applied by the NASA Goddard Space Flight Center for evaluating defects in their projects. In [BCR94] the authors motivate the use of the GQM approach in the context of software engineering as a measurement instrument. The base assumption of this approach is that before defining metrics, the goals must be clear. A GQM model is therefore developed goal-oriented, i.e., top-down starting with defining the set of quality and productivity goals, then going on by formulating questions for each goal and finally associating metrics to the questions. A goal definition should contain the purpose of research, the issue to study, the object to analyze and a chosen viewpoint (project manager, customer, etc.).

The major problem of case studies is how to compare projects. The costs of carrying out the same project twice, one time with the technology to examine, another time without it, are unacceptable high. A feasible approach is to use similar projects, where comparable goals were achieved with similar staff using the same development methods with specific changes under study, and compare relevant parts. In particular, this is of interest for organizations with projects mainly comprising their base knowledge (same technology, same development methods) which conduct case studies frequently and therefore collect much material for comparing projects. For more information on case studies see [FLP02].

2.3 Alternatives to Case Studies

Another method to gain knowledge about projects is Post Mortem Analysis. In contrast to case studies, PMAs are conducted after the project is finished. This approach has some advantages over case studies, but also some disadvantages. One advantage is that there is no influence of the research activities on the (running) project, so one has not to take care to eliminate this kind of disturbance later. Another advantage is the lower budget needed for PMAs. The main disadvantage of PMAs is the fact that for analysis there is only material produced for the project by the project. Important information about conflicts between project members, problems with the technology and solutions for those problems, etc. may not be available anymore, there was no need to document it. Case studies are conducted in parallel to the projects and can overcome this problem by documenting important facts. For more information on PMA see [BDS02]. The guidelines given in [KPP+02] help avoiding sources of error often found in empirical research.

3 Case Studies in Software Migration Projects

The goal of the project M-WISE (Modelling of knowledge-intensive processes in software engineering) is to encourage and improve the use of knowledge management techniques in software engineering. One part in the project is to conduct external case studies in soft-

ware migration projects of our industrial partners PSIPENTA and altavier. The knowledge modeling description language KMDL [GW04] is employed to model software development processes and to evaluate effects of this approach on those processes. Additionally, we perform surveys on role-based software engineering and on the use of knowledge management in software development enterprises. To validate the hypotheses examined in the case studies, controlled experiments are applied. Two case studies are conducted at the two software developing companies in the context of migration.

We identified several typical problems which have to be addressed when conducting a migration project. One typical problem which should be evaluated is the elicitation of interfaces between components or systems. Thus our first hypothesis H_1 is:

H_1 (informal): Eliciting the interfaces between legacy components or systems is more difficult than just defining new interfaces.

A formalized variation of this rather unconcrete and informal hypothesis could be:

H_1: The total costs of eliciting the interfaces between legacy components or systems are much higher than the costs of just defining new interfaces.

In this case the used metric is *total costs* which is directly measurable. Hypotheses for other typical problems of migration projects can be formulated similarly.

A very important issue seems to be the used programming paradigma or programming language. Today's programmers often have problems to understand code written in old-fashioned languages like COBOL or Fortran using non-object-oriented programming paradigms. More organizational issues focus on the number of components which have to be migrated, their interfaces and size. Typically, those facts are needed to determine the number, scale and scope of migration steps. Is it easier to do many tiny steps changing only bits of the system or is it better to do larger steps, sometimes accidently resulting in a big-bang strategy? Another scope is the technique of redocumenting software using inspections. It will be useful to test whether those inspections have more impact on the quality of a system than traditional methods only using the provided documentation or code comments. Based on the results gained from evaluating those and more factors within the case study, it is possible to learn how knowledge management techniques can improve the processes.

More general information on software migration is available in [BS95]. The hypotheses to quantify in both surveys have been concretized. Obviously the most interesting questions in both cases are whether and how knowledge management techniques can be helpful for software development processes, in which project phases they are most reasonable and how significantly the effects of its usage can improve software development by means of the above mentioned problems. After reformulating this rather abstract study goal in measurable hypotheses, we defined objective (e.g. "How many requests per hour can be processed") and subjective (e.g. "How satisfied are the employees with the software system") metrics for each question.

4 Conclusion

Case studies and PMA are different ways for software development organizations to get scientifically verified analysis about new technologies or organizational development processes in their specific context. However, each approach has its specific benefits and limitations in an industrial setting. In any case, scientifically validated information is a valuable assistance for decision-making opposed to personal experience which is most often rather limited in its applicability to software engineering processes. The KMDL is used to check elicited generic processes for weak spot analysis. A comparison between actual process and proposed process provided starting points for constructing metrics for the GQM approach which can be used to track the improvement. Addressed issues have been presented in the context of this paper, most important issues cover processes of elicitating interfaces, the effect of the use of certain programming languages and the sizes of migration steps. The metrics will provide a starting point on how the migration process actually can be improved by using knowledge management technology and how the organization can learn from former migration projects.

Literatur

[BCR94] Victor Basili, Gianluigi Caldiera und Dieter Rombach. *The Goal Question Metric Approach*. Wiley, 1994.

[BDS02] Andreas Birk, Torgeir Dingsoyr und Tor Stalhane. Postmortem: Never leave a project without it. *IEEE Software: Special issue on Knowledge Management*, 19(3):43–45, 2002.

[BS95] Michael L. Brodie und Michael Stonebraker. *Migrating Legacy Systems: Gateways, Interfaces & the Incremental Approach*. Morgan Kaufmann Pub, 1995.

[FLP02] Bernd Freimut, Oliver Laitenberger und Teade Punters. *Tutorial: Empirical Studies in Software Engineering*. Fraunhofer Institut Experimentelles Software Engineering, 2002.

[Gla04] Robert Glass. Pilot Studies: What, Why, and How. *Journal of Systems and Software*, 36:85–97, 2004.

[GW04] Norbert Gronau und Edzard Weber. Defining an infrastructure for knowledge intensive business processes. In K. Tochtermann und H. Maurer, Hrsg., *Proceedings of I-Know 2004*, Seiten 424–431. Springer, 2004.

[Kar01] Richard Karash. Dialog on Learning Organizations. *http://www.learning-org.com/*, 1994-2001.

[KPP+02] B. Kitchenham, S. Pfleeger, L. Pickard, P. Jones, D. Hoaglin, K. Emam und J. Rosenberg. Preliminary Guidelines for Empirical Research in Software Engineering, 2002.

[Rob99] Pierre N. Robillard. The Role of Knowledge in Software Development. *Communications of the ACM*, 42(1):87–92, 1999.

[Tiw04] Amrit Tiwana. An empirical study of the effect of knowledge integration on software development performance. *Information and Software Technology*, 2004.