Scandinavian Journal of Information Systems

Volume 22 | Issue 1 Article 2

Summer 6-30-2010

Making Sense of Software Project Management: A case of knowledge sharing in software development

Annemette Kjaergaard
Copenhagen Business School, amk.inf@cbs.dk

Peter Axel Nielsen
pan@cs.aau.dk, pan@cs.aau.dk

Karlheinz Kautz
Copenhagen Business School, karlheinz.kautz@rmit.edu.au

Making Sense of Software Project Management

A case of knowledge sharing in software development

Annemette Kjærgaard
Copenhagen Business School, Denmark
amk.inf@cbs.dk

Peter Axel Nielsen
Aalborg University, Denmark
pan@cs.aau.dk

Karlheinz Kautz
Copenhagen Business School, Denmark
khk.inf@cbs.dk

Abstract: In this paper we empirically explore knowledge sharing in a group of project managers in the Danish software company SpaceSoft. We apply a framework of sensemaking that focuses on how people participate in creating shared meanings. The framework is used in the analysis of the case where project managers create shared knowledge in a handbook for software project management. The framework provides a rich conception of how meaning is created. It explains the importance of collisions and negotiation of the project managers' expectations and experience. The findings add to existing theories of knowledge sharing in software development. We contribute in particular with an in depth explanation of the complex process where personal knowledge gradually turns into shared knowledge and some of it in codified form becomes part of the software project management handbook.

Keywords: Knowledge sharing, software development, project management, sensemaking.

Accepting editor: Rikard Lindgren

© Scandinavian Journal of Information Systems, 2010, 22(1), 3–26

1 Introduction

Our area of concern in this article is knowledge sharing in software development, which has been of interest to IS researchers for quite a while. We describe the different strands of research on the topic and identify a lack of empirical research of how knowledge is shared in software companies and in particular how process descriptions are developed and used by software developers and managers. Our objective is to contribute to this omission. Our primary focus is on how people share knowledge in software development and we want to open the black box of knowledge sharing to understand how it occurs, how developers and managers create shared understandings, and which role process descriptions play in this context. Such research is valuable for developers and managers when they plan and engage in knowledge sharing and relate to process descriptions.

For this purpose we empirically explore knowledge sharing among a group of project managers in the Danish software development company SpaceSoft. SpaceSoft is a mature organization, which develops software as a subcontractor for the European Space Agency (ESA). We studied the creation of a project management handbook as part of a large action research project. The purpose of the project management handbook was to improve the capability of Space Soft's software projects to meet customers' requirements, to follow ESA's standards, and to deliver software within budget and time.

For the analysis of the empirical data we use a framework which understands knowledge sharing as an ongoing collective organizational sensemaking process (Kjærgaard and Kautz 2008). This approach focuses on collisions and negotiation of the involved actors' expectations and experience. The framework allows us to investigate of our research questions which are: How do project managers draw on past experience when they share knowledge and collectively contribute to process descriptions in a project management handbook? And how do they collectively make sense of the project management handbook's contents and use?

The paper is structured as follows. Section 2 identifies the existing research on knowledge sharing to improve software development. In section 3 we present the theoretical framework we use for analysing the empirical data. The research design is described in section 4 and the case company is introduced in section 5. Section 6 holds the analysis and our findings. These are then discussed in section 7 and in section 8 we conclude the article.

2 Knowledge sharing to improve software development

The challenges in software development are vast, and considerable research has addressed how these challenges can be met by passing on experience, hard earned knowledge, and well-proven practices to other software developers and managers (Walz et al. 1993). This has, in particular, been studied from a knowledge sharing perspective. The research has so far led to two distinctly different and almost disjointed research paths: the technical and the social.

Along the technical path we find the experience factory (Basili et al. 1994; Basili and Caldiera 1995) advocating that a designated organization takes experiences from one project and transfer it to other projects. Successful application of the experience factory idea has been reported by, e.g., Basili and Green (1994), Lindvall et al. (2001), and Komi-Sirviö et al. (2002). According to these authors the experience factory requires a separate organization and laborious knowledge elicitation, but they do not mention the costs of establishing and maintaining the necessary organization, structures and processes. In a parallel effort Althoff et al. (2000a, b) have addressed computer-support for the experience factory. In their approach they build experience bases with codified knowledge from which they extract lessons to be learned (see also von Wangenheim et al. 2000). One particular form of experience base is the post mortem report written at the completion of each development project (Pedersen 2005; Kasi et al. 2008). Several research reports, however, suggest that it is not common to perform post mortem analyses of projects. Desouza et al. (2005) suggest that the benefits are not matching the costs. They put forward that highly structured reports have low costs, but are difficult to use later, while there are high costs of reporting incurred in writing a useful story, which is however easier to use later. Schalken et al. (2006) present a case study and argue that traditional post mortem reports must be transformed into quantitative factors to enable statistical analysis of the post mortem data. Kasi et al. (2008) report that the barriers to post mortem reporting are: (1) the organizational context; (2) the focusing of the effort and the data collection; (3) analysis of the collected data; and (4) sharing and exploiting the findings.

There has been much discussion on how knowledge processes in general can be supported by IT (Tsoukas 1998; Hansen et al. 1999; Swan et al. 1999; Alavi and Leidner 2001). A technological view on knowledge sharing provides, however, a much too limited view on individual and organizational knowledge processes (Scarbrough et al. 1999). In the field of software development Liebowitz (2002) reports on a successful application of a computer-based system for knowledge sharing, but not without much overhead in capturing and codifying knowledge. Liebowitz and Megbolube (2003) have studied the complexity of developing and of using different knowledge management systems. Codified knowledge that is placed in repositories of best practices and process descriptions they classify as having medium development complexity and low to medium use complexity. In studies of successful software process improvement where processes have been described and used, these descriptions are often also available on the company intranet (e.g., Pries-Heje et al. 2008).

Along the social path we find applications of knowledge management theories to software process improvement. Based on social theories such as Nonaka's knowledge creation theory (Nonaka 1994; Nonaka and Takeuchi 1995) several studies acknowledge how difficult it is to move knowledge between its tacit and its explicit dimension and between the individual and the group: in software process improvement (Arent and Nørbjerg 2000; Kautz and Thaysen 2001; Mathiassen et al. 2002); in software process maturity (Baskerville and Pries-Heje 1999; Ravichandran and Rai 2003); and in software process implementation (Mathiassen and Pourkomeylian 2003; Meehan and Richardson 2002; Nielsen and Tjørnehøj 2010). Software process improvement is largely occupied with descriptions of software development processes, how they are produced, what their content should be, how they are assessed, and how they are implemented (e.g., Humphrey 1989). Knowledge management theories have so far been applied to explain why it is so difficult to move back and forth between software developers' experience and competence in terms of personal and tacit knowledge and process descriptions as a form of shared and explicit knowledge of development teams or organizations. Little has so far been achieved to bridge this gap. Some conclude that the idea of process descriptions should be abandoned (Aaen 2003), some, who are mostly advocating for the Capability Maturity Model, argue that the process descriptions are more important than software developers and managers (e.g. Humphrey 1989; Chrissis et al. 2003), and others are seeking a middle ground where process descriptions have a role to play when utilized by software developers and managers to mediate sharing of knowledge (e.g., Mathiassen and Pourkomeylian 2003; Nørbjerg et al. 2006; Hosbond and Nielsen 2008; Kautz and Hansen 2008). Knowledge sharing processes are not widely practised and Desouza (2003) claims that the necessary knowledge about software development cannot be captured and codified and that personalized knowledge is perhaps more effective. In a study of 72 case studies in the European Software Process Improvement database, Fehér and Gábor (2006) have found that 65% of the studied software companies were "sharing and reusing [...] existing knowledge' while as few as 26% were 'developing and creating knowledge about processes."

In summary, research along the technical path misses the important social dimension of understanding knowledge sharing in software development. So far, research along the social path has been limited and with little appreciation of the role that process descriptions plays in knowledge sharing. On this background we investigate how knowledge is shared in software companies and in particular how process descriptions are developed and used from a social perspective.

3 Theoretical framework

As a theoretical framework for our study we use a process model for understanding knowledge sharing as organizational sensemaking. The process model is based on a theoretical model, originally developed and used to explain the establishment of a knowledge management initiative in a technology friendly organizational environment (Kjærgaard and Kautz 2008). The model combines Burgelman's (1983; 2002) framework of corporate venturing with Weick's (1979; 1995) work on sensemaking and meaning construction and provides an understanding of how users' cognitive frames influence, as well as are influenced by, cognitive processes. The model combines the theories of Weick and Burgelman in order to provide a more detailed view on the cognitive processes for strategic action as a consequence of a changed frame of reference in the process negotiating shared meaning. The model is illustrated in figure 1. The following description of the framework is based on (Kjærgaard and Kautz 2008).

The two sub-processes, creating and negotiating, each have a set of cognitive processes as well as a dominant construed reality, which represents the organizational members' dominant cognitive frame of reference. Construed realities are collectively held perceptions of how to behave. A construed reality is an assembly of more or less connected pieces of information and beliefs, which together form a picture that confirms or constructs a reality. Furthermore, a triggering event of collision between expectations and experiences marks the transition from one sub-process to the other. Although there is no specific indicator for time, the change from one stage to the other indicates that the process unfolds over time.

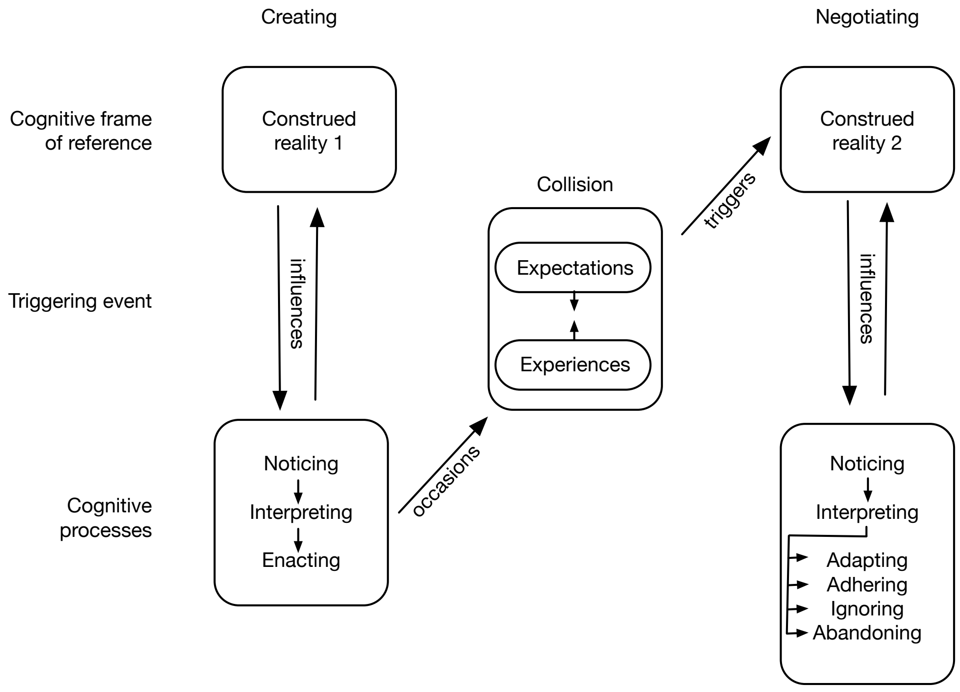

Figure 1: Processes of organizational sensemaking (Kjærgaard and Kautz 2008)

The construed reality constitutes the cognitive frame of reference, which forms part of the meaning construction equation in sensemaking. A frame in sensemaking presents an overall paradigm or shared understanding to which cues are related to create meaning. Frames tend to be past moments of socialization, which relate present moments of experience as cues to create meaning. According to Weick meaning is created if a person can construct a relation between past and present moments: "the content of sensemaking is to be found in the frames and categories that summarize past experience, in the cues and labels that snare specifics of present experience, and in the ways these two settings of experience are connected" (Weick 1995, p. 111).

In the model only the dominated construed reality is shown as the cognitive frame in each of the sub-processes of creating and negotiating. However, more construed realities can co-exist. The model also emphasizes the cognitive processes at the individual level, which form the individual organizational members' sensemaking and meaning construction. These processes are influenced by the collective construed realities of the organization.

3.1 Creating

The cognitive processes in the first period of creating are based on the understanding that organizational members notice in particular the parts of the ongoing flow of information that they are exposed to. This sensemaking is the effort to create meaningful action based on beliefs and expectations. Beliefs are found in ideologies, cultures, scripts, and traditions. Beliefs and expectations are directional in their operations and filter the input, thereby guiding what is being noticed. They are building blocks for the construction of objects, which are subsequently noticed as real. According to (Weick 1995, p. 152), "[w]hen perceivers act on their expectations, they may enact what they predict will be there. And when they see what they have enacted, using their predictions as a lens, they often confirm their prediction." In this respect, sensemaking is as much about plausibility and coherence as it is about accuracy.

What organizational members bracket out of the ongoing flow in the creating process is based on the dominant construed reality, which guides their noticing. They *interpret* the noticed information drawing from the construed reality, which they finally enact. The following process of *enacting* confirms and strengthens the construed reality. This cyclic process can be seen as a process of thinking in circles or what Weick (1979) would refer to as a self-fulfilling prophecy.

3.2 Collision

Intermediating the two sub-processes is a collision between expectations and experiences. It is the organizational members' perception of dissonance between expectations and experiences that creates uncertainty and *triggers* them to pursue stability by establishing a new frame into which their experiences fit. This collision between expectations and experiences is an *occasion* for sensemaking, which sets off an intense process of meaning construction in the form of a negotiation where the organizational members reinterpret the dominant construed reality or create anew.

Although sensemaking is a continuous process, it can intensify on several occasions. There are two types of occasions, which can cause a sufficient shock, and thereby initiate an instance of sensemaking in organizations: ambiguity and uncertainty. Both types of occasion create an interruption in an ongoing flow, although the shock is different in the two types. Ambiguity is the situation where "the assumptions necessary for rational decision making are not met" (Weick 1995). The problem here is not that information is insufficient, but that more information may not resolve misunderstandings. Uncertainty, in contrast, governs when there is a lack of knowledge, which might thus be resolved by gaining additional information.

Basically any kind of experience can serve as an occasion for sensemaking as long as it is unexpected. Some occasions initiate unconscious sensemaking, and some initiate conscious sensemaking. In both cases it is important to understand what happens in the sensemaking process as this has implications for the subsequent meaning construction, which guides further action and sensemaking of a situation.

3.3 Negotiating

The cognitive processes of negotiating are more ambiguous than those of creating as they are influenced by a new dominating construed reality, which has not yet necessarily replaced the construed reality of the creating process. In the negotiating process, sensemaking is based on action as well as beliefs. The organizational members themselves create the action, which guides the negotiation and which leads to sensemaking as committing. By committing an action, the action is made irrevocable and thus more difficult to change than the beliefs about that action. Commitment thus imposes a form of logic on the interpretation of action. Commitment reduces flexibility, learning and adaptation because it makes withdrawal difficult, and commitment thus slows adaptation to change. On the other hand, commitment makes it easier to get things done, as it focuses the social construction of reality on those actions, which are high in choice, visibility and irrevocability.

Depending on which of the construed realities dominate, the organizational members create meaning out of their experiences and subsequently act upon this meaning. The cognitive processes in the negotiating thus reflect the extent to which the actors *adapt*, *adhere*, *ignore* or *abandon* their beliefs and actions.

One possible reaction is that members *adapt* their beliefs to the new construed reality and consequently adjust their activities to better fit this new situation. Adaptation in this situation is a classic learning situation where members' experience becomes encoded into the new construed reality as coming to terms with the incongruence between expectations and experience. Learning can be seen as having taken place by the updating of their view of reality and as a result the new dominating reality can be formalized.

Alternatively, the organizational members *adhere* to the former construed reality dominating the creating process. A dominant element of the negotiating sub-process is the actions themselves. The organizational members make sense of a situation that they have created themselves and which results in further commitment to their own actions. Adherence might change over time as negotiations continue.

A third response is to *ignore* experiences and simply try again. Where the processes of adapting and adhering both entail continuation of the action generated in the creating or negotiating process, the process of ignoring entails discontinuation of this action. Members make sense of their lack of success in negotiating a new construed reality by, for example, blaming their own inadequacy. They still draw from the first dominating construed reality, but discontinue the accompanying action. Also ignoring might change through a continuing negotiation sub-process.

Finally, the organizational members may accept the new reality and *abandon* the action, which does not fit the new construed reality. By doing so they accept the change and the fact that previously proposed ideas do not fit any longer. Abandoning subsequently also allows for a formalization of the new construed reality.

4 Research design

The research process falls into two related parts firmly linked through a collaborative practice research (CPR) approach as outlined by Mathiassen (2002): the first part follows an action research approach, and the second part is a practice study.

CPR, in which researchers and practitioners closely collaborate, is organized as action research complemented with experiments and practice studies. The action research was conducted in a Danish software company, SpaceSoft, which develops software for the European Space Agency. The action research served a dual purpose (McKay and Marshall 2001): practical problem solving and contribution to research. In SpaceSoft the problem solving purpose was to improve software processes following the lines described in (Mathiassen 2002), i.e., (1) diagnose the problems with the current software processes and practice; (2) design how to improve software processes and change software development; (3) take action to change accordingly; and finally (4) assess how well the improvements have been and then re-start a cyclic improvement process. The research purpose was to investigate the application of knowledge management theory in software process improvement, i.e., to study in which ways the practical problem solving could be supported by applying a knowledge management approach. The CPR approach was particularly appropriate as the researchers worked with SpaceSoft's CEO and its project managers within a project setup that had the same structure as the project approach reported in (Mathiassen 2002).

The practice study was conducted within the realm of CPR and resembled what Braa and Vidgen (1999) call an action case, which is a mixed approach of action research and case study research. More precisely, the practice study was conducted on the backdrop of the action research. It was based on the action research criterion that an improvement effort should be evaluated. The empirical data concerning the creation of the shared handbook for software project management had been collected throughout the action research, but the analysis of the data was performed in retrospect using the sensemaking framework (Kjærgaard and Kautz 2008) introduced in section 3. The framework was not used in action and therefore it is prudent to state that the analysis is more a result of the retrospective analysis of the action research than a direct result of the action research itself. With regard to the discussion of criteria for action research (e.g., Nielsen 2007) and the issue whether a framework should be declared in advance of a study to guide the action as suggested by Checkland (1991), our study has been performed from the perspective that an analytical framework to understand the actions and their outcome does not have to be declared in advance and it does not have to be useful in action. We have instead utilized the framework to understand the actions and the results after the fact because they did not make much sense to us while in the midst of the action.

The study took place, as mentioned above, in SpaceSoft, a Danish software company. The company and a group of four action researchers collaborated over a period of more than two years to improve software processes. The research reported in this article spanned a period of 12 months, but also included a larger contextual study and other improvement projects for a duration of more than two years. The overall project was planned and progress was monitored through monthly 1-day meetings between the action researchers, the CEO and 4-6 project managers.

The researchers initially set up the collaboration. The identification of areas to be improved had been a joint effort, but the company had decided all the improvement efforts. One of the improvement efforts that had been started by the company, with the CEO in particular as an eager participant, was the development of a project management handbook. The intention was to improve SpaceSoft's project management competencies and to support knowledge sharing amongst the project managers.

This particular improvement effort ran for more than 12 months and a first complete version of the handbook was presented and reviewed after eight months. Figure 2 depicts the timeline of the improvement effort. The numbers below the arrow state the elapsed time. Activity 'M' was the 2-day start-up meeting of the effort. Activities labelled 'W' represent individual writing and

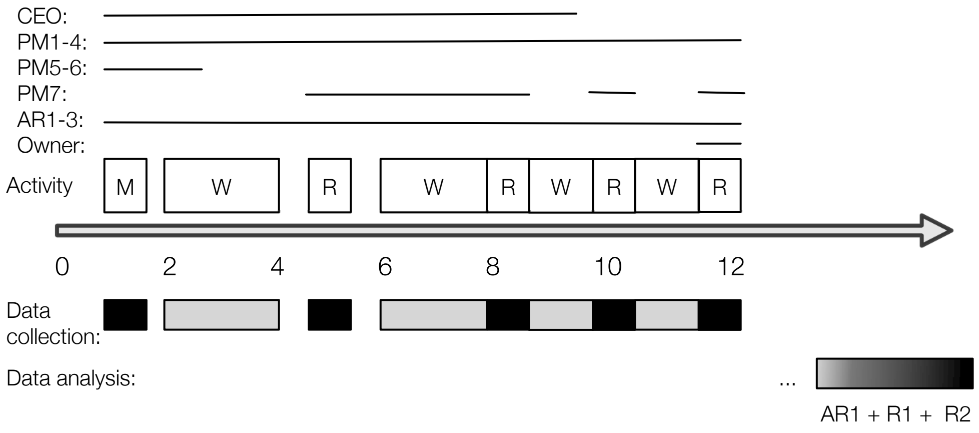

Figure 2: The timeline of the collaborative practice research during the development of the project management handbook

re-writing periods. Activities marked as 'R' were full-day review sessions; after month 10 they became half-day sessions. The top line signifies the activities, in which in the CEO was active, the lines below show the same for the seven project managers, the three action researchers, and the owner of the company. The line labelled 'data collection' below the arrow signifies the time periods when data collection was intense (black) and when it was occasional (grey). The last line indicates that the data analysis for the research documented here was performed much later by one of the involved action researchers (AR 1) and two additional researchers (R1 and R2). Their roles are explained below.

The action researchers collected empirical data during the meetings and sessions described in the timeline and through the action researchers' active participation in reviewing the whole handbook and in writing parts of it. It comprised field notes, email correspondence, meeting minutes and process documentation in a versioned sequence of increasingly complete project management handbooks and reviews hereof. The active participation took place during several meetings between the CEO, the involved project managers, and the action researchers as outlined in the timeline. During these meetings the action researchers collected data by participating and observing while taking notes. Moreover, data from the larger action research project enabled a fuller understanding of the context, what was discussed at the meetings as well as of different participants' views on the process and the other participants' viewpoints.

After the completion of the data collection in the action research process we analyzed the case. Our data analysis went through four steps. In the first step we constructed the above time line of the project. In the second step two of the researchers, one of the action researcher, who had participated in the project, and a second researcher, who had been involved in an earlier application of the framework, reviewed the data. We discovered that the overall course of the project matched the process of organizational sensemaking as described in (Kjærgaard and Kautz 2008). However, we also recognized that we needed additional data. Therefore in a third step we extended our data and included the author of the framework (Kjærgaard 2004) in the research team. The additional data came about from the latter two authors' audio-recorded interviews and interrogations of the action researcher in 3 interview sessions. In the fourth step the three researchers then mapped the sensemaking process of the case onto the process model of organizational sensemaking in an iterative process comprising a further 3 sessions. The three researchers discussed every discrepancy between the case and the model in the mapping at length and in detail until it could be resolved. In every instance where a significant interpretation of the case did not immediately fit well into the framework, great care was taken not to force the case and our interpretation onto the framework. The combination of intervention, interpretation and collaboration between the three researchers with different levels of involvement introduced new analytical perspectives on the case and brought the interpretive rigour that resulted in our understanding of the creation of the software project management handbook that we describe in the next sections.

5 Case study – SpaceSoft

SpaceSoft produces software as a subcontractor for the European Space Agency. It develops a wide range of dedicated software for on-board, micro-gravity, verification and validation, ground station control, and checkout systems.

The company was founded in 1992 and is rather old in the fast moving software development business, and it has lived through many changes. Most of the company's software developers have a M.Sc. in engineering or computer science and have developed software for the space industry for many years and are quite experienced with the particulars of space products. A perhaps equally large number of the software developers have little experience within the space industry, but rely on experience from other domains of software development.

In 2002 SpaceSoft decided to focus on improving its software processes and entered into a collaborative research project called Software Processes and Knowledge. The initiative to improve software processes started with a traditional, though light-weight, assessment of the current software processes, based on the understanding of the action researchers that improvement should be initiated with an assessment (Mathiassen et al. 2002) though not necessarily with a formal and model-based one (Iversen et al. 1999). SpaceSoft's current software processes at that time were compared informally to the processes in the Bootstrap software capability model (Kuvaja 1999). The results showed significant discrepancies between the Bootstrap model and the company's current software processes and practices. This led to a decision to prioritize improvement of requirements engineering and project management though other processes were also in need of improvement. The focus of this article is on how SpaceSoft addressed the improvement of project management. The improvement activities outlined in figure 2 in the context of the research design are in the following described in more detail:

- Activity M in month 1: A 2-day start-up meeting with the CEO, 6 project managers and 2 action researchers. The action researchers contributed with a theoretical overview of project management. The project managers contributed with presentations of current practices and problems in their project management as well as with problems regarding how to interpret ESA's guidelines. The meeting ended with decisions on the content of the handbook and who of the project managers should write which process descriptions. The effort also became organized as a project with its own project plan, goals, and schedule.

- Activity W in months 2-4: Individual writing of chapters for the handbook. During this period there were occasional email correspondences between some of the project managers and the action researchers on more theoretical issues of project management. During this period two project managers left the project.
- Activity R in month 5: A 1-day review and discussion of the first draft of the handbook. Participants in this meeting were 3 action researchers, the CEO, the four remaining project managers from the first seminar, and a newly hired project manager who joined the project.
- Activity W in months 6-7: Individual re-writing of chapters for the handbook. The CEO and the now five project managers all participated in the writing. The action researchers were used occasionally as consultants by a few of the project managers.
- Activity R in month 8: A 1-day review and discussion of the second draft, which comprised all chapters outlined in the project plan for this improvement effort. In this meeting 3 action researchers, the CEO, and five project managers participated.
- Activity W in month 9: Individual re-writing of chapters for the handbook. Four of the five project managers participated in the writing of chapters.
- Activity R in month 10: A half-day review and discussion meeting with participation of two action researchers and five project managers.
- Activity W in month 11: Individual re-writing of chapters for the handbook. Four of the five project managers participated in the writing of chapters.
- Month 12: A half-day final review of the handbook. Two action researchers, one of the owners, and five project managers participated in this meeting. It ended with an acceptance of the handbook subject to some minor changes.

ESA has a large number of standards, with which its subcontractors including SpaceSoft must comply. A number of these are process standards and SpaceSoft has in the past dealt with the issues of compliance uniquely in each development project. The ESA standards are complex. They form a hierarchy of standards have considerable variation in levels of detail in instructions and whether instructions are required, recommended, or optional. Thus, the documentation of compliance is never trivial and requires project managers to be well-read in the many ESA standards. This led to the idea for improvement, namely that a new and improved project management process should be documented in a handbook. The handbook should be compliant with the relevant ESA standards and it was expected that it would be much easier to document the compliance when project management was performed in accordance with the guidelines in the handbook.

It is SpaceSoft's declared strategy to deliver fixed-priced software on time. The CEO in particular expressed a deep concern for achieving this goal and was very clear in maintaining that all improvement activities should be directed at this. There was a strong belief among some of the experienced project managers and the CEO that the ESA standards reflected the best practices within the discipline and the software space industry and that—as they already applied a number of these best practices—SpaceSoft was in compliance with the ESA standards. Thus, their initial idea was that the software process descriptions for software project management, which should be documented in the handbook, had to encompass these already existing practices. The experienced managers were convinced that they already knew what to do; it was just not documented in a shared handbook.

Other project managers were not as optimistic about the existing practices and wondered whether they were in fact performing best practices. Their view was that not all the project managers have the necessary training in these processes; they did not necessarily have the competence needed, and therefore some training and education would be necessary at least for new project managers. This led to the idea that a project manager education should be established based on the new and improved processes.

While these differing perspectives stood out quite clearly in retrospect, they were not that obvious at the beginning of the improvement project as they gradually emerged during the process of creating the project management handbook. This is described and discussed in more detail in section 6.

As described in the timeline above, the project management improvement project started with a series of workshops with the purpose of: (1) designing and writing a handbook for project management, and (2) designing an education or training course on the contents and use of the handbook. In the workshops, experienced as well as inexperienced project managers participated together with the CEO and two to three action researchers.

During the first two-day meeting, the project managers worked in small groups to define what topics the handbook should cover and how the handbook should be written. Although the participants had quite different ideas, opinions and experiences, the discussions were conducted in a friendly atmosphere. The result of the meeting was a list of topics that should be covered in the handbook. The work on each topic was organized as a handbook chapter to be written and assigned to one or two project managers, who were expected to find the time to write first draft descriptions as an additional part of their work assignments.

The identification of the topics was to a large extent based on the project managers' experience and only to some extent came from the ESA standards. Some of the processes were easily written by one or two project managers and never led to any controversy in later workshops. A few significant processes were very difficult for the project managers to write. Although it was a design goal for the handbook to be brief, they eventually drafted quite complex process descriptions, which varied widely in form and content. These writings did not for the most part meet the expectations of the other project managers and were as a consequence heavily criticized by those at later workshops. This led to a number of iterations over the process descriptions, which were expanded and condensed several times while the core ideas of the processes were continually negotiated during the review workshops.

The handbook was designed, written and reviewed through these workshops, but the aim of the workshops changed during the progress of the project and attention was diverted to related problems as well. The project manager training-course was never designed, and the project management handbook gradually expanded its scope to become a handbook for software engineering processes as well. The review workshops were never chaotic, but there was disagreement about many issues. Most disagreements were overcome and problems were solved. However, a fundamental issue arose and gradually caught the attention of the participants during the last workshops. This issue was a concern for where the project managers' knowledge came from and how they could properly share this knowledge.

This issue arose from the participants' experience that there were two groups of processes: (a) the 'easy processes', which were easy because the project managers and the CEO shared the knowledge necessary to perform and describe the processes, and (b) the 'difficult processes', which were difficult because the participants did not share the necessary knowledge for writing the procedures. A few of the project managers and the CEO in particular were of the opinion that the handbook should be written and formalized and that all the project managers should then perform the formalized processes based on the instructions in the handbook. Several other project managers had the view that the handbook needed to contain more than a set of instructions; it should also contain explanations, which would enable a project manager to perform the processes. In this situation, where experiences and expectations collided, the project managers entered a negotiation process, which finally resulted in a handbook that was accepted by all project managers and consequently used in SpaceSoft.

To analyze the differences and their resolution in more detail, we now use the framework of knowledge sharing as sensemaking that was introduced in section 3.

6 Analysis

We first present three examples of content from the software project management handbook and then present an overall analysis of the creation and negotiation of the handbook content. The three examples are: a technique for estimation, a template for project planning, and a procedure for shipment. They show that the processes of creating and negotiating the handbook content were necessary for illuminating the two groups of project managers' quite different perceptions of project management and the acceptance of the different actions resulting from the negotiation of the meaning of the new handbook.

6.1 A technique for estimation

Prior to the creation of the handbook, the usual practice of estimation at SpaceSoft was an educated guess, qualified by the project managers' own experience. The estimate was subsequently presented to the CEO who, based on his experience and the interests of the company, discussed it with the respective project manager and made adjustments before approving it. Typically it was in the interest of the project manager to get as much time allotted to the project as possible, while top management aimed at spending as few resources as possible in order to win the contract and to ensure a reasonable profit.

When the first draft of the estimation technique was presented, a huge debate arose about how to estimate projects. The draft was written by one of the more inexperienced project managers, who had noticed the arbitrary estimation process at SpaceSoft and had interpreted the lack of standards as uncertainty about how to conduct state-of-the-art estimation. He enacted his belief in project management as a profession, which would benefit from more theory on new practices in the draft.

The draft included several important concepts of estimation, presented a number of techniques and ended by promoting a three-point estimation that was then explained in some detail. The draft was written with the intention of explaining estimation techniques also for those who had not previously seen nor applied estimation techniques, and it resembled a section of a textbook.

The project manager's past experience varied considerably concerning how they estimated projects and tasks and also in their success rate. The draft was briefer than a standard textbook on software engineering, but it covered similar contents. The most experienced project managers noticed the level of detail in the draft and interpreted it as distrust in their ability to carry out what they had done successfully many times. During the discussion there was a strong feeling among some of the project managers that these techniques would not improve the estimation accuracy, but also that the draft was either too detailed for those who already knew the techniques or too abstract for those who did not. After a long discussion where the collision between the two groups' expectations and experiences was explicated, the author of that draft section was persuaded to rewrite it.

At a later workshop, a second draft of the estimation technique was discussed. This draft contained only the recommended three-point estimation and a brief terminology on different estimates used by both ESA and SpaceSoft. It was evident that for new project managers the project management handbook could not substitute training and education in estimation, but it did provide an introduction to central concepts of estimation specifically focusing on ESA's definitions in order to ease communication with ESA's project managers.

At the final approval of this section of the handbook, the project managers drew from a shared construed reality where project management was not reduced to a set of standard procedures to follow, but was seen as a skill to be performed by knowledgeable individuals, who would benefit from having their attention directed toward what they already knew, but might have forgotten in a stressful context.

6.2 A template for a project plan

It was a widespread conception in SpaceSoft that project planning is a difficult discipline—primarily because plans were continuously changing or ought to be changed, but were not. Reasons for not making required changes to project plans were typically shortage of time when new requirements were accepted or when project managers did not realize that (or how) new requirements changed the course of an assignment's implementation. At other times it was because they had not made detailed arrangements with ESA and consequently did not have a comprehensive understanding of the extent of the changes. The variety of project plans was significant. ESA did not require a particular way, nor did SpaceSoft internally, which led to a multiplicity of solutions. The differences between the various project plans could not be explained by the differences between the projects' conditions and contents and it was thus somewhat a matter of personal choice and taste which type of project plan was used.

In the process of creating the handbook, there was a heartfelt wish among the project managers that the handbook should address the issue of project planning. This wish was strongly supported by the CEO, who wanted the project plan to be viewed as a contract between him and the project manager, which then could form the basis for more detailed discussions of the project at the beginning as well as during the project.

A less experienced project manager wrote the first draft of this chapter. He had noticed problems with project planning and had interpreted the issue of project planning as something that could only be done properly if described in detail. This was enacted in the draft as a procedural project planning description. In his opinion, the purpose of the handbook chapter on project planning was to create a standard project planning procedure, which would be relevant for top as well as project management by making it possible to ensure that all necessary elements were included and thereby would guarantee a higher degree of comparability across projects.

However, the first draft was met with reluctance. The more experienced project managers found that the procedure was not detailed enough to make a comprehensive project plan and furthermore questioned whether it was at all possible to create an exhaustive procedure for project planning. The CEO was specifically critical, as he did not find anything in the new draft that he could use to prevent the project managers from making 'bad' plans. Although a lot was included in the chapter about how, for example, to make an initial work breakdown structure and the proposed standard plan also included the task of developing such a structure; there was nothing about how a project plan developed dynamically over time. This did not fit the CEO's principle of having a detailed plan for the next three days, more general plans for the next three weeks and very general plans for the next three months. From his perspective a plan was a dynamic tool, which needed to be updated continuously.

As a consequence of this collision, the issue of project planning was discussed and negotiated intensively among the project managers and a new understanding of what attention the issues should be given in the handbook was created. This new understanding was reflected in the second draft, which was more "to the point" according to the project managers. Instead of describing project planning as a detailed procedure to follow, it now comprised a template including a few important principles for good project planning. The template in its final form stipulated headings and contents of the project plan, but it also included which states the plan had to go through (e.g., draft, approved) and how the project manager and the CEO would discuss monitoring of progress. By the end of the last review workshop there was agreement among the project managers that the template was highly relevant and that they would adapt their work to it accordingly.

6.3 A procedure for shipment

There had been several incidents in the past where projects had not been able to deliver products to the customer on time or with items missing due to problems with packaging and shipment. Based on this experience, the CEO suggested including a procedure in the handbook, which would ensure, in detail that all possible problems in the final shipping process were taken care of in due time and he made a first draft of this procedure himself. The procedure included very detailed instructions, e.g., on shipping of hardware in wooden boxes if necessary, on how to obtain customs clearance and on the assembly process. The project managers were rather sceptical of the CEO's proposal as they found it much too detailed and in sharp contrast to other parts of the handbook, where many details had been left out.

The procedure was thus discussed at the workshops where in particular the level of detail was the focus. The CEO argued for a detailed procedure drawing from his experience, in which shipments did not live up to even simple requirements of packaging, were late without good reason or were incomplete. Although another group of project managers agreed that failures occurred, they found it unnecessary to instigate a strict procedure, as they feared that this would make the shipment process take up too many resources and potentially make the process even slower. In their opinion, shipment was a necessity, but not a core competence and they did not want to waste too much time on it. Through his experience with unsatisfied customers who complained about missing items or late arrivals of products, the CEO perceived shipping as a very important procedure as this was the final contact between SpaceSoft and its customers. His reasoning was that a bad experience might leave the customer with a bad impression of SpaceSoft regardless of the quality of its products or services.

The project managers were divided into two opposing groups in the discussion of the shipping procedure: those who believed that a procedure with the level of detail proposed would be too tedious to follow, and those who were of the opinion that it was a necessity to have this level of detail in what they believed to be a very important process. The procedure was continually discussed at several workshops and a new shared interpretation of the procedure emerged as a checklist rather than a recipe for action. This interpretation as a checklist made sense to both groups and was accepted by everyone as the proponents of the detailed procedure argued that it would ensure that mistakes and omissions were avoided while the critics argued that not every step should be executed if it was not relevant in a specific shipping context.

6.4 The process of creating and negotiating the content of the handbook

In the first period of creating the project management handbook, two different construed realities were in play. One construed reality was that of the CEO and a smaller group of project managers, who saw the handbook as the means for ensuring that project managers followed standards and stuck to agreements. They shared the view that project management requires good skills, including the ability to manage deliverables and keep deadlines, and that project management is not a theoretical discipline. They all had a history as successful project managers and even though some of them had experienced budget deficits, they were convinced that this could be avoided by following best practice for project management. Failure to comply with standards, not meeting deadlines and failing quality of deliverables were in their opinion caused by sloppiness and carelessness. They preferred the content of the handbook to be concise and brief, reflecting existing practice and primarily being a mutual agreement among the project managers to carry out their jobs according to their existing knowledge of how to manage projects well. What they noticed in the creating process was that the project managers in general carried out project management tasks by drawing from previous experience. They interpreted this as a confirmation that project managers already knew how to perform their tasks. Following on from this, they enacted this belief by suggesting that the project handbook should reflect existing best practice.

Another construed reality coexisted and dominated another group of project managers' views of the handbook in the creating process. This second construed reality was deeply rooted in the group's day-to-day experience with project management as a constant battle. Part of this experience included projects that had been late, had changing requirements negotiated with ESA managers, and had difficulties of delivering products at the agreed time with the agreed features. To these project managers it did not make sense to simply stick to plans and follow procedures. In cases of problems in projects they rarely experienced that the problems could be alleviated by adhering to the plan or by re-planning. To them the problems were usually of the kind that they had not anticipated, they had unknowingly allowed the ESA managers to introduce the problem or a technical issue had proved to be a much bigger risk than foreseen. These project managers believed in planning, but only to a limited extent. In their noticing and interpreting of events and problems, they relied therefore only to some degree on plans and procedures, because rigidly following processes did not make sense to them. What they enacted, when developing the handbook, was therefore a much more flexible view of project management as a constantly changing process, which could not be controlled by plans and procedures. Instead they suggested that project managers, and particularly new project managers, needed to be trained in project management. The role of the handbook was in their view to summarize and communicate the knowledge necessary to understand tasks in project management, as well as different techniques and methods available to solve the tasks. Moreover they emphasized that the handbook should form the basis for a course on software project management and not stand alone.

During the first workshops both construed realities influenced the discussions and the development of the handbook. The different ways of interpreting and enacting the contents of the handbook led to much confusion in discussions that could not immediately be reconciled. Applying the framework of organizational sensemaking it becomes apparent that in the common wish to develop the handbook, the participants experienced a collision between how the two groups of project managers perceived software project management.

The actual occasion for the collision was the writing of the draft chapters. Although there was general agreement about what topics should be covered in the handbook, the content of each topic was up for serious discussion and members of the one group criticized chapters written by members of the other group. Whereas one group argued that long and detailed explanations were irrelevant for a handbook, the other group found the brief and concise descriptions too 'thin' to make a useful contribution. The two groups had expected variations in opinions about the content of the handbook, but they were surprised that their views were so different. The collision between the expectations and experiences of the two groups was quite extensive and it took several meetings to unravel what the differences consisted of.

Only gradually did this collision lead the project managers and action researchers to realize that the views embedded in the project management handbook needed to be negotiated. During the negotiation process the two groups slowly constructed a new, and this time, shared construed reality.

The negotiation process took time and involved numerous iterations of the handbook chapters. There was a high degree of uncertainty concerning the final result. The chapters stayed with the same author during the rewriting process and each chapter had to go through several rounds of at times harsh criticism before it had a format that members from both groups could agree upon. The process was slow partly because no extra resources were provided by management and partly because the authors felt de-motivated by the continuous demand for rewriting their contributions. However, their efforts to approach one another proved valuable in creating a better understanding of project management. By continually discussing the content of the chapters, the members gained a better understanding of their different approaches to project management. This triggered the emergence of a new shared construed reality.

The three examples presented above showed the negotiating process in more detail. However, the most important result of the negotiating process was the handbook itself, which was completed and is now in use by all project managers in SpaceSoft.

All four cognitive processes of negotiating – adapting, adhering, abandoning and ignoring—were in play during the process, but ultimately adapting became dominant over time. Some project managers initially adhered to their ideas and found it difficult to change their view of project management, and others intermittently even abandoned being part of the process of creating the handbook or ignored that project management was an ambiguous issue. However, in the end all project managers accepted the project management handbook as the binding document that comprised their shared knowledge.

7 Discussion

The analysis of the case shows that the development of the handbook was not just a simple matter of externalizing the project managers' individual and tacit knowledge. We have used the framework as a theoretical lens to explore knowledge sharing in software development and can summarise the analysis in the following three findings:

1. The project managers' dominant construed realities of the issues in question are essential for understanding how they make sense of these issues. Their different construed realities lead to a collision between their expectations and experiences.
2. The collisions are important and should be accepted because they trigger negotiations of a new construed reality.
3. The negotiation process is necessary for the project managers to work on their different construed realities. The negotiations should not be terminated prematurely for the short-term sake of time and efficiency, but at the risk of becoming ineffective on the long term. Resolving collisions takes time and resources.

The handbook was therefore not a compromise or a reflection of one group's dominant knowledge of project management but a product of a negotiation process in which a new construed reality was constructed. Our findings point to a critique of the view of knowledge that is held by the supporters of the technical path of research into knowledge sharing in software development. In this line of research, knowledge is often reduced to a question of what should be stored in a database and how the data in the base should be searched, cf. section 2 (e.g., Basili et al. 1994; Rus and Lindvall 2002). Based on our analysis we concur with the critique of this technical view. Desouza et al. (2005) e.g., suggest that the costs of producing post mortem reports are much higher than their benefits. This resembles our findings: The creation of a process description requires a rather high cost, but the benefits of negotiating it to a level where project managers adapt it are much higher and outweigh these costs. However while post mortem reports are mere evaluations, a negotiated process description sets direction for future action. We further suggest that building experience bases which contain codified knowledge (e.g., Althoff et al. 2000a, b; von Wangenheim et al. 2000) is very costly, but will only have a limited effect. The contents of repositories, from which knowledge and lessons learnt can be extracted, is unlikely to make sense to the project managers who have not been involved in its negotiation.

Our findings support instead the social research path on knowledge sharing in software development, as outlined in section 2. There are several research reports, which take a balanced and social view on the matter. First, there are the studies, which show that the tacit dimension of improving software processes is significant and cannot be reduced in its complexity (Arent and Nørbjerg 2000; Kautz and Thaysen 2001; Mathiassen and Pourkomeylian 2003). Our analysis concurs with these studies, which are largely based on Nonaka's knowledge creation theory (Nonaka 1994; Nonaka and Takeuchi 1995). We have here used an elaborate form of sensemaking as our theoretical lens (cf. section 3) and reached similar results, but from a different theoretical perspective and, thus, extend these previous results in the following way: According to Nonaka's knowledge creation theory externalization is the process for transforming tacit knowledge into explicit knowledge. Our study shows in detail how the transformation process cannot simply be understood as externalization, but has to be understood as a consequence of the collisions of experiences and expectations that happen over time. In particular, the complex negotiation is not a matter of extracting knowledge from the deep and dark places where habitual behaviour rests. Rather, it is a social process where different project managers' views and beliefs meet. This sometimes leads to new and shared knowledge—and sometimes not.

Second, we offer a perspective on knowledge sharing among software project managers with a detailed description of how experiences, expectations, and in particular negotiation are part of sensemaking. Our literature review of knowledge sharing in software development in section 2 is comprehensive and we have not come across a documented case study where the process of constructing a process description (here a handbook for project management) has been described in such detail. Our approach allows a deeper understanding of what it takes for a company such as SpaceSoft to write and document a software process and then seek to adapt to it. Based on (Hansen et al. 1999) Mathiassen and Pourkomeylian (2003) explore in a case study the balance between a codification strategy and a personalization strategy, that is, the balance between, on the one hand, codifying existing knowledge, storing and reusing it through the use of IT, and on the other hand, perceiving knowledge as intrinsically individual and personal to be shared when people interact for example in communities of practice. They conclude that to improve software development, a knowledge sharing strategy is needed that strikes a balance between codification and personalization. Our study adds deeper insight into how such a balance is achieved and how personalized knowledge of software project management becomes codified knowledge of software project management through negotiation and how it through adaptation becomes personalized again. Furthermore, we also show how other elements of organizational sensemaking such as continual collisions, as well as the adhering, ignoring, and abandoning of construed realities delimit this process. The knowledge processes in SpaceSoft were highly personalized, to the extent where knowledge sharing between project managers had become very time consuming. Hence, they decided to move in the direction of a more codified approach to knowledge sharing. Our study therefore also illustrates what it takes for a software company to change its knowledge sharing strategy. It takes collisions and negotiations of a new construed reality.

Third, it has been suggested in (Kautz and Hansen 2008) that knowledge sharing can be supported by mapping flows of knowledge in software development and by utilizing these maps in a diagnosis of knowledge flow problems. We suggest that while this approach may be useful for understanding the operational sharing of knowledge over a long period of time and in a stable state our contribution is a different one. We have given a detailed account of how a group of project managers share knowledge during a change of the knowledge sharing process. Our account emphasizes the negotiating of a new construed reality. In this new construed reality it may make sense to analyze the flow of knowledge, but we suggest that the mapping could be extended by also taking into account the sensemaking processes, to, among others, see how adapting, adhering, ignoring, and abandoning construed realities are played out over time in a software company.

Fourth, it is a core idea of software process improvement that software processes should be written or otherwise documented (Humphrey 1989) in order to reduce a software company's dependency on employing the right people. In this view, which is a genuine codification strategy of knowledge for software development (Mathiassen and Pourkomeylian 2003), processes are in this view more important than people. Our findings relate to this understanding of software process improvement in two ways: (1) The case study shows that there is a limit to how well such processes can provide independence of knowledgeable people, i.e., the project managers. The process descriptions result from a change process, which involves a negotiation of their contents. The process descriptions' usefulness and acceptance will depend on which people participate in the negotiation process and over a longer period of time it will also depend on how many of these people remain in the company to maintain the negotiated results. If the agreement falls apart due to too many new people or other external circumstances the company will be back at a state where several construed realities exist. (2) The case study also shows that the level of detail in process descriptions varies and has to be negotiated. The idea that there are 'best practices', which will fit all software companies, managers and developers is questionable as descriptions of 'best practices', which have not been produced through negotiation will only achieve a limited acceptance.

Fifth, our research shows that the sensemaking framework provides an alternative perspective. The framework underlines the importance of focusing on the process of creating and negotiating construed realities as a basis for a process description instead of on the process description itself. The project management handbook as a solution was only useful and significant because the processes of creating and negotiating it were in focus and considered useful by the involved project managers. This does not necessarily mean that initiatives to support knowledge sharing in software development without involving project managers or developers are not valuable. We argue, however, that initiatives, which do not involve project managers or developers, might face a high risk of not producing usable results and they may thus fail. Focusing explicitly on the sensemaking process by imposing on the participants to spend time on the creation and negotiation of meaning was resource demanding, but it produced useful results.

8 Conclusion

In this article we addressed the research questions of how software project managers draw on past experience when they collectively contribute to a process description, in this case a software project management handbook, and how they collectively make sense of its contents and use. The presented exploratory case study, which we analysed with a framework for organizational sensemaking, contributes to a better understanding of the phenomenon. This understanding can be summarised as follows:

1. The project managers' construed realities are essential for the creation of a process description.
2. Collisions of their expectations and experiences are important and should be accepted.
3. Negotiation of the process description is necessary.
4. As a result the process description become a shared product, which the project managers can adapt to.

Our case analysis shows that by applying the sensemaking framework (Kjærgaard and Kautz 2008), a better understanding of the complexity involved in creating a shared process description for software development is provided. This is in contrast to the technical path of knowledge sharing theory in software development. It coincides well with the social research path, which it also extends by shedding more light on how this path may be followed in practice.

9 References

Alavi, M., and Leidner, D. E., (2001). Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundation and Research Issues. *MIS Quarterly*, (25:1): 107-136.

Althoff, K.-D., Bomarius, F., and Tautz, C., (2000a). Knowledge Management for Building Learning Software Organizations. *Information Systems Frontiers*, (2:3-4): 349-367.

Althoff, K. D., Müller, W., Nick, M., and Snoek, B., (2000b). KM-PEB: An Online Experience Base on Knowledge Management Technology. In: *Advances in Case-Based Reasoning. Proc. of the 5th European Workshop on Case-Based Reasoning (EWCBR'00)*, E. Blanzieri and L. Portinale, editors, Springer, Berlin.

Arent, J., and Nørbjerg, J., (2000). Software Process Improvement as Organizational Knowledge Creation—A multiple case analysis. In: *Proceedings of the 33rd Hawaii International Conference on System Sciences*, Wailea, Hawaii, p. 11.

Basili, V., and Caldiera, G., (1995). Improve Software Quality by Reusing Knowledge and Experience. *Sloan Management Review*, (37:1): 55-64.

Basili, V., Caldiera, G., and Rombach, H. D., (1994). The Experience Factory. *Encyclopedia of Software Engineering*): 469-476.

Basili, V., and Green, S., (1994). Software Process Evolution at the SEL. *IEEE Software*, (11:4): 58-66.

Baskerville, R., and Pries-Heje, J., (1999). Grounded Action Research: a Method for Understanding IT in Practice. *Accounting, Management and Information Technologies*, (9:1): 1-23.

Braa, K., and Vidgen, R., (1999). Interpretation, Intervention and Reduction in the Organizational Laboratory: A Framework for In-context Information Systems Research. *Accounting, Management and Information Technologies*, (9:1): 25-47.

Burgelman, R. A., (1983). A Process Model of Internal Corporate Venturing in the Diversified Major Firm. *Administrative Science Quarterly*, (28:2): 223-244.

Burgelman, R. A., (2002). *Strategy is Destiny. How Strategy-Making Shapes a Company's Future*, The Free Press, New York.

Checkland, P., (1991). From Framework through Experience to Learning: The Essential Nature of Action Research. In: *Information Systems Research: Contemporary Approaches and Emergent Traditions*, H.-E. Nissen, H. K. Klein and R. A. Hirschheim, editors, Amsterdam, North-Holland, pp. 397-403.

Chrissis, M. B., Konrad, M., and Shrum, S., (2003). *CMMI: Guidelines for Process Integration and Product Improvement*, Addison Wesley.

Desouza, K., Dingsøyr, T., and Awazu, Y., (2005). Experience with Conducting Project Postmortems: Reports versus Stories. *Software Process Improvement and Practice*, (10): 203-215.

Desouza, K. C., (2003). Barriers to Effective Use of Knowledge Management Systems in Software Engineering. *Communication of the ACM*, (46:1): 99-101.

Fehér, P., and Gábor, A., (2006). The Role of Knowledge Management Supporters in Software Development Companies. *Software Process Improvement and Practice*, (11): 251-260.

Hansen, M. T., Nohria, N., and Tierney, T., (1999). What's Your Strategy for Managing Knowledge? *Harvard Business Review*, (77:2): 108-116.

Hosbond, J. H., and Nielsen, P. A., (2008). Software Process Improvement Using Light Knowledge Tools. In: *Software Processes & Knowledge: Beyond Conventional Software Process Improvement,* P. A. Nielsen and K. Kautz, editors, Software Innovation Publisher, Aalborg, pp. 103-116.

Humphrey, W., (1989). *Managing the Software Process*, Addison-Wesley Publishing, Reading.

Iversen, J., Nielsen, P. A., and Nørbjerg, J., (1999). Situated Assessment of Problems in Software Development. *The DATABASE for Advances in Information Systems*, (30:2): 66-81.

Kasi, V., Keil, M., Mathiassen, L., and Pedersen, K., (2008). The post mortem paradox: a Delphi study of IT specialist perceptions. *European Journal of Information Systems*, (17:1): 62–78.

Kautz, K., and Hansen, B. H., (2008). Mapping Knowledge Flows. In: *Software Processes & Knowledge: Beyond Conventional Software Process Improvement,* P. A. Nielsen and K. Kautz, editors, Software Innovation Publisher, Aalborg, pp. 89-102.

Kautz, K., and Thaysen, K., (2001). Knowledge, Learning and IT Support in a Small Software Company. *The Journal of Knowledge Management*, (5:4): 349-357.

Kjærgaard, A., (2004). Knowledge Management as Internal Corporate Venturing: A Field Study of the Rise and Fall of a Bottom-Up Process, Copenhagen Business School, Copenhagen.

Kjærgaard, A. L., and Kautz, K., (2008). A Process Model of Establishing Knowledge Management: Insights from a Longitudinal Field Study. *OMEGA*, (36:2): 282-297.

Komi-Sirviö, S., Mäntyniemi, A., and Seppänen, V., (2002). Towards a Practical Solution for Capturing Knowledge for Software Projects. *IEEE Software*, (May/June): 60-62.

Kuvaja, P., (1999). Bootstrap 3.0—A SPICE1 Conformant Software Process Assessment Methodology. *Software Quality Journal*, (8): 7-19.

Liebowitz, J., (2002). A Look at NASA Goddard Space Flight Center's Knowledge Management Initiatives. *IEEE Software*, (May/June): 40-42.

Liebowitz, J., and Megbolube, I., (2003). A Set of Frameworks to Aid the Project Manager in Conceptualizing and Implementing Knowledge Management Initiatives. *International Journal of Project Management*, (21): 189-198.

Lindvall, M., Frey, M., Costa, P., and Tesoriero, R., (2001). Lessons Learned about Structuring and Describing Experience for Three Experience Bases. In: *Learning Software Organizations (LSO)*, K.-D. Althoff, R. L. Feldmann and W. Müller, editors, Springer Lecture Notes in Computer Science, LNCS 2176, pp. 106-119.

Mathiassen, L., (2002). Collaborative Practice Research. *Information Technology & People*, (15:4): 321-345.

Mathiassen, L., Nielsen, P. A., and Pries-Heje, J., (2002a). Learning SPI in Practice. In: *The Agile Software Development Series*, L. Mathiassen, J. Pries-Heje and O. Ngwenyama, editors, Addison-Wesley, Boston, pp. 3-21.

Mathiassen, L., and Pourkomeylian, P., (2003). Managing Knowledge in a Software Organization. *Journal of Knowledge Management*, (7:2): 63–80.

Mathiassen, L., Pries-Heje, J., and Ngwenyama, O., editors, (2002b). *Improving Software Organizations: From Principles to Practice*, Reading, MA, Addison-Wesley.

McKay, J., and Marshall, P., (2001). The Dual Imperatives af Action Research. *Information Technology & People*, (14:1): 46-59.

Meehan, B., and Richardson, I., (2002). Identification of Software Process Knowledge Management. *Software Process Improvement and Practice*, (7:2): 47-55.

Nielsen, P. A., (2007). IS Action Research and Its Criteria. In: *Information Systems Action Research: Bridging the Industry-University Technology Gap*, J. N. F. Kock, editor, Springer Verlag, New York.

Nielsen, P. A., and Tjørnehøj, G., (2010). Social Networks Analysis in Software Process Improvement. *Software Process Improvement and Practice*, (22:1): 33-51.

Nonaka, I., (1994). A Dynamic Theory of Organizational Knowledge Creation. *Organization Science*, (5:1): 14-37.

Nonaka, I., and Takeuchi, H., (1995). *The Knowledge-Creating Company*, Oxford University Press, Oxford.

Nørbjerg, J., Elisberg, T., and Pries-Heje, J., (2006). Experiences from Using Knowledge Networks for Sustaining Software Process Improvement. In: *Proceedings of the Eighth International Workshop on Learning Software Organizations,* Rio de Janerio, Brazil. pp. 9-17.

Pedersen, K., (2005). Managing Learning in Systems Development Projects. Ph.D. thesis, Department of Computer Science, Aalborg University.

Pries-Heje, J., Nørbjerg, J., Aaen, I., and Elisberg, T., (2008). The Road to High Maturity: How the first Danish company reached CMMI level 5 in 100 months. In: In: *Software Processes & Knowledge: Beyond Conventional Software Process Improvement,* P. A. Nielsen and K. Kautz, editors, Software Innovation Publisher, Aalborg, pp. 163-191.

Pries-Heje, J., and Pourkomeylian, P., (2004). Beyond Software Process Improvement: A Case Study on Change and Knowledge Management. In: *EuroSPI Conference Proceedings*, Trondheim, Norway.

Ravichandran, T., and Rai, A., (2003). Structural analysis of the impact of knowledge creation and knowledge embedding on software process capability. *IEEE Transactions on Engineering Management,* , (50:3): 270–284.

Rus, I., and Lindvall, M., (2002). Knowledge Management in Software Engineering. *IEEE Software*, (19:3): 26-38.

Scarbrough, H., Swan, J., and Preston, J. (1999). Knowledge Management: A Literature Review, Institute of Personnel and Development, London.

Schalken, J., Brinkkemper, S., and van Vliet, H., (2006). A Method to Draw Lessons from Project Postmortem Databases. *Software Process Improvement and Practice*, (11): 35-46.

Swan, J., Scarbrough, H., and Preston, J., (1999). KM – The Next Fad to Forget People. In: *Proceedings of the 7th European Conference on Information Systems*, J. Pries-Heje, C. U. Ciborra, K. Kautz, J. Valor, E. Christiaanse, D. Avison and C. Heje, editors, Copenhagen.

Tsoukas, H., (1998). The Word and the World: A Critique of Representationalism in Management Research. *International Journal of Public Administration*, (21:5): 781-817.

Walz, D. B., Elam, J. J., and Curtis, B., (1993). Inside a Software Design Team: Knowledge Acquisition, Sharing, and Integration. *Communications of the ACM*, (36:10): 63-77.

von Wangenheim, C. G., Althoff, K. D., and Barcia, R. M., (2000). Goal-oriented and similarity-based retrieval of software engineering experienceware. In: *Learning Software Organizations, Lecture Notes in Computer Science,* LNCS 1756, Springer, Berlin, pp. 118-141.

Weick, K. E., (1979). *The Social Psychology of Organizing*,2nd ed., McGraw-Hill, New York.

Weick, K. E., (1995). *Sensemaking in Organizations*, Sage, Thousand Oaks, CA.

Aaen, I., (2003). Software Process Improvement: Blueprints versus Recipes. *IEEE Software*, (20:5): 86-93.