Drifting Software Process Improvement: Studying Practice

Tjørnehøj, Gitte

Publication date:
2009

Document Version
Accepted author manuscript, peer reviewed version

Ph.D. Thesis

Drifting Software Process Improvement: Studying Practice

by

Gitte Tjørnehøj

Make everything as simple as possible
…but not simpler.

Albert Einstein

This thesis has been submitted and defended at
the Faculties of Engineering, Science and Medicine,
Aalborg University, Denmark
in partial fulfillment of the requirements for
the Ph.D. degree in Computer Science.

The defense took place on December 21th 2010.

Opponents

Ivan Aaen, Associate Professor (chairman)
Department of Computer Science, Aalborg

Karlheinz Kautz, Professor
Copenhagen Business School, Copenhagen

Peter Carstensen, Forsknings- og innovationschef
Alexandra Instituttet, Copenhagen

Supervisor

Peter Axel Nielsen, Professor
Department of Computer Science, Aalborg University

Abstract

Software process improvement (SPI) has, since the late eighties been a common framing when improving system development practice. The field was sparked when Humphrey published his work on managing the software process. The field of SPI is an applied research field and most contributions are either prescriptive or descriptive, resulting from a kind of learning loop between the industry and the organizations developing the Capability Maturity Model or other similar models. The field lacks independent and reflective research.

This PhD study takes its outset in doubt about whether the theory of the SPI field actually expresses an appropriate understanding of the practice of the field. This doubt is based partly on personal practical experience in the field and partly on the fact that more and more problems and failures are reported from attempts to adopt SPI technology. The aim of this study has therefore been to study SPI practice in depth in order to characterize SPI practice and shed light on whether SPI theory is consistent with SPI practice.

The study was organized as part of a collaborative practice research project and involved a literature study, an action research intervention and a longitudinal interpretive single case study. The results are presented in this PhD thesis, based on four papers published during the PhD project.

The main result is that SPI practice is characterized by drifting SPI technology. Plans are made, control is exercised, but SPI technology drifts in unpredictable directions anyhow. Looking further into this I found important conditions for drifting SPI practice. First the SPI network is inherently dependent on the production network in the software organization. Second the adoption of SPI technology is by nature longitudinal and sensitive towards dynamic environments. The complexity and dynamics that this imposes on SPI practice become beneficial if embraced. This can be done by negotiating the adoption of SPI technology between control and drift. One important aspect of this negotiation is to cultivate the organization for improvisational action. The characteristics found for SPI practice challenge existing SPI theory.

The implications of the characteristics challenging SPI theory is that we need to reinterpret the existing SPI theory in the light of a much more profound understanding of the complexities of SPI practice. We need to explore radical new ways to deal with improving practice. Practitioners will have to leave a controlling strategy, in order to negotiate control and drift when adopting SPI technologies.

This thesis provides further details on the research project, approaches and results. The thesis consists of four journal papers and this summary at hand.

KEYWORDS: SPI PRACTICE, CONTROL AND DRIFT, IMPROVISATION

Resumé

This is a Danish translation of the abstract.

Siden firserne har software procesforbedring (SPI) været den mest anvendte teoretiske ramme for forbedringer af systemudviklingspraksis. SPI opstod da Humphrey publicerede sit arbejde om ledelse af software processer. Forskningen i SPI er præget af præskriptive og deskriptive forskningsbidrag, der stammer fra en slags udviklingscyklus mellem industrien og de organisationer, der udvikler de førende modeller indenfor området, særligt the Capability Maturity Model (CMM). Forskningsfeltet udviser en mangel på uafhængig og reflekterende forskning.

Dette ph.d.-studie er udsprunget af en undren over, hvorvidt den opfattelse af SPI praksis, der præger forskningen i SPI feltet, faktisk afspejler den reelle praksis. En undren som dels bygger på egen praksis erfaring, og dels er støttet af, at nye forskningsbidrag i stigende grad peger på problemer med indførelsen af SPI teknologi i virksomhederne. Formålet med studiet har derfor været, gennem detaljerede studier af SPI praksis at belyse om den eksisterende SPI teori er konsistent med den nuværende SPI praksis. Arbejdet har omfattet et litteraturstudie, en aktionsforskningsintervention og et fortolkende longitudinalt casestudie.

Hovedresultatet er at SPI-praksis ikke som forventet er rationelt planlagt og implementeret, men generelt præges af store afvigelser, udsving og uforudsete forandringer i processen. Dette kan med et engelsk fagligt begreb benævnes ”drift”[1]. Jeg har identificeret nogle vigtige grunde til at SPI-praksis ”drifter”: For det første er SPI-netværket i en software virksomhed totalt afhængigt af dennes produktionsnetværk. For det andet er indførelsen af SPI teknologi af natur en longitudinal proces, der er sensibel overfor forandringer i omgivelserne. Den dynamik og kompleksitet som dette tilfører SPI-praksis kan vendes til gavn for virksomheden, hvis det udnyttes på passende vis. Man bør afveje anvendelsen af kontrol og ”drift” mekanismer fleksibelt i indførelsen af SPI teknologi, og kan med fordel bevidst udvikle virksomhedens evne til at improvisere.

De nævnte egenskaber ved SPI-praksis viser sig at anfægte den eksisterende SPI-teori. Som konsekvens bør vi genfortolke den eksisterende SPI-teori i lyset af en meget dybere forståelse af kompleksiteten af SPI-praksis. Vi bør også udforske og afprøve radikalt anderledes måder at udføre SPI-praksis. Endeligt må SPI-praktikere forlade de ensidede kontrol-baserede tilgange til indførelse af SPI-teknologi til fordel for at afveje kontrol og ”drift” i dette arbejde.

Forskningen og dens resultater uddybes i denne sammenfatning der er baseret på fire publicerede artikler.

EMNEORD: SPI-PRAKSIS, KONTROL OG ”DRIFT”, IMPROVISATION

[1] Min anvendelse af begrebet bygger på Claudio Ciborras arbejde (Ciborra, 2002).

Acknowledgements

During the last six years I have been re-trained from the industry to academia through my work with this thesis and through my teaching. A lot of people have supported, inspired, taught and encouraged me in this process and I am grateful for all the help.

Most importantly I want to thank Lars Mathiassen. It all started when he invited me to leave the software industry and start a PhD study. He has been my mentor all through - though not supervisor. I have learned so much trough collaborating with him and he hosted me and my family during a wonderful and educational posting in Atlanta.

I want to thank all my colleagues in the IS group and the participating researchers of SPK research project for letting me be their apprentice in academic practice.
I will especially thank Jan Pries-Heje for reminding me how far I had to go, Jeremy Rose for teaching me my very first lessons on English academic writing, Karl Kautz for introducing me to Scandinavian research tradition, Ivan Aaen for insisting on agility and Jacob Nørbjerg for guidance on praxis studies.

I also want to thank all the PhD students for inspirational discussions and for sharing fun and frustrations. (The SPI-ce girls: Anna Börjesson and Galina Hansen, Bo Hansen Hansen, Thomas Elisberg, Keld Pedersen, Jens Henrik Hosbond, Rune Høegh, Janne Jul Jensen, John Persson, Rolf Njor Jensen, Lise Hermansen, Karsten Jahn and Fred Durao).

The most important grounding for my research is my practice experience. All my former colleagues in the software industry and at the Business School have all contributed to these invaluable and enjoyable experiences. I have gained most of my knowledge through practice. A special thank to the case firm *SmallSoft* and all the employees there. Without your positive collaboration this research would not have been possible.

Last but not least I will thank Peter Axel Nielsen for supervising me through six years of studies. It has been a long journey and I may from time to time not have been the easiest PhD student to supervise. However this thesis proves that he succeeded in his efforts.

Henning and Simon have been my closest and most precious travelling companions on my PhD journey. They may not always understand my incentives and priorities, but they always respect my decisions. I am thankful for their support and especially for their decision to follow me on my posting in Atlanta.

Table of Contents

1 Introduction

This thesis has from the start been influenced significantly by the fact that I was a practitioner before entering academia. My effort has been focused on results that are useful in practice. I was inspired by my experience with improvement of system development practice gathered through 10 years working in the Danish software industry. This chapter presents my research goals and topic based on my motivation and lays out the structure of the thesis.

1.1 Personal motivation

I have practiced a broad range of the activities commonly involved in system development; software engineering, management and quality assurance. I have had rich opportunities to be involved in improvement efforts both as target for the improvements and as designer and implementer of the improvements.

The first improvement effort I participated in was an attempt to achieve an ISO9001 certificate (Hoyle, 2005). We employed a decentralized approach involving most system developers in designing, testing and implementing new procedures. It was engaging and interesting to participate, but it was also time consuming *and* sometimes it turned into a battle between colleagues. The first improvements were rather easy to agree upon and to implement with good results, but we increasingly found the changes required by the norm were less helpful and more difficult to design and implement. Obviously this was because we had started with the changes that could immediately ease our work or that were requested by our customers. Much of the new procedures required more overhead work in documenting and some of the required changes were even perceived to be destructive to our flexibility and creativity. After a couple of years the strategy of achieving a certificate was ditched as it became clear that a certificate was not required to stay in the market. Also, we had realized how costly and difficult the improvement was. However, the software firm kept a full quality assurance organization working since we found that the quality assurance effort in many ways had proved to be beneficial. Our long term improvement approach emphasized real and sustainable improvements over a full set of procedures (according to the standard).

The Capability Maturity Model (CMM) (Paulk et al., 1993 and section 2.2) was introduced at a later date, when the organization was involved in a research project with university students experimenting with mini assessments according to CMM. The assessment was followed by some initial improvement activities. We found the CMM experiment interesting, especially the new and tempting concept of measurement, but we did not change our improvement strategy, i.e., the quality assurance organization of the firm. I personally was a bit sceptical toward CMM and thought it to be overly detailed and inflexible. How could this prescribe processes for practice? In my view, system development practice demanded flexibility and situated methods. I also found the assessment rather simplistic and was dubious about how helpful it had been.

Improving system development practice was indeed important for the firm to stay competitive and to provide interesting jobs for employees. The continued improvement effort was convincingly beneficial in many ways, but the improvement technologies had to be adapted to the needs of the firm. Working with the two norm driven approaches; the ISO9001 (Hoyle, 2005) and CMM (Paulk et al., 1993), left me with the impression that they were difficult to comply with and that they could turn out to be inappropriate for an organization. They did not sufficiently meet the improvement needs of Danish software firms. Improving system development in practice turned out differently than prescribed in the approaches.

These experiences of designing and implementing improvement efforts according to the main approaches have been my personal motivation for researching improvement practice.

1.2 Research goals

My research has been guided by two equally important goals. The goals are rather general, but they have helped me focus both the research question and the approach of my research to be able to provide feasible results.

In the first goal I emphasize *relevance of my work to the software industry*. Improving system development practice for a software organization is crucial to stay competitive. Being a former practitioner simply makes it important for me to provide useful results for the industry.

Thus I want to

- Contribute relevant knowledge to the software industry in order to support their efforts to improve their system development practice. (Research goal 1)

This means that my research topic and results need to be relevant and useful for practitioners, and the results need to be published in a form that suits this particular audience.

My second research goal emphasizes *independent and reflective* research. My research was initiated by a literature study of the software process improvement (SPI) research field, which demonstrated a lack of reflective and independent research (Hansen et al., 2004a, section 5.2). Thus I want to

- Contribute to the research field of software process improvement with independent and reflective research. (Research goal 2)

This means that my research approach has to provide for reflections on SPI topics and that my research organization should strive to keep me independent of firm or other interest.

The first goal pushed practice to the center of my research. This is visible in the research questions, approach and organization. The second goal led to the in-depth research approach. Reflective independent research can help dig deeper into everyday SPI practice to understand and explain the complexity of it.

The two goals have an important link, as more profound understanding can lay the basis for better advice to practice (Mathiassen, 2002).

1.3 Research topic: SPI practice

In Scandinavia IS research has to a large extent been focused on studying system development practice and targeting how to improve that practice. Mathiassen described the historical evolution of research in the field in three areas of system development challenges (Mathiassen, 1997). In the first area, before the mid seventies, the improvement efforts were focused on methods, tools and project management. During the next 15 years, quality assurance and CASE technologies was found increasingly interesting, before software process improvement (SPI) (Humphrey, 1989) attracted the most attention in the late eighties.

The introduction of the Capability Maturity Model (CMM) (Humphrey, 1989, Paulk et al., 1993) to the field of software development introduced the concepts of software processes and maturity as the key concepts of any improvement efforts. The maturity model included most of the challenges of system development that had been in focus before, and added more. It introduced a priority of the challenges to deal with first (the levels), and provided the software process concept as the one way to describe, control and manage improvements of system development practice. Since SPI has been the main challenge of improving system development in this latest time period, this study is framed as a SPI study.

My practice experience (section 1.1) is from around the time when the focus shift towards SPI had reached the Danish software industry and the Danish research community.

My first research goal is "providing relevant knowledge to the software industry in order to support their efforts to improve their system development practice". If I should rephrase this in the framing of SPI it could become: I want to provide useful knowledge to SPI practitioners of the software industry.

Practice implies in general that theories are brought into use situated in a context of work, people and organizations and their context. Or phrased the other way around: Practice is when people work in organisations and may bring theory in use. Being a practitioner, the last phrasing makes most sense. In either case practice in this context means SPI practice and involves mainly SPI practitioners and their work, system development work, software engineers, software organizations and their environment.

1.4 Structure of the thesis

Chapters 2 and 3 present the theoretical background for the PhD study. Chapters 4 and 5 describe the research project and the contributions. In chapters 6 and 7 I discuss and conclude the work.

First, *chapter 2* introduces the research topic of SPI as being a field with its origin in CMM (Paulk et al., 1993) and the thoughts presented by Humphrey in "Managing the software process" (1989). I describe the special challenges that small and medium-sized firms face when adopting SPI technology before I focus on my SPI research background as being part of the Danish research on SPI.

Second, *chapter 3* presents the reference theories used in my study. Claudio Ciborras' view on the adoption of technology as governed by drifting forces (Ciborra, 2002) has been the main theory helping me to interpret my findings from SPI practice. It was supplemented by the theory of organizational improvisation (Cunha et al., 1999) when investigating drifting in more details.

C*hapter 4* describes the research design of the thesis. The research methods used in the study are presented and longitudinal interpretive case studies are discussed as an appropriate approach. The collaborative practice research project that provided my research organization is briefly presented, before I describe the resulting research design in more detail. The four journal papers that contribute to the thesis are presented briefly in *chapter 5*, describing the research approaches, findings, results and contributions.

In *chapter 6* I discuss SPI practice on the basis of the findings. The discussion is organized in five themes. For each theme the evidence from my work is recapitulated and traditional SPI theory is discussed in accordance with the theme.

The theme discussion is concluded with a formulation of my contributions against the backdrop of other research contributions. I sum up by discussing the answers to my research questions and the implications of my findings for SPI research and practice.

C*hapter 7* concludes the PhD thesis by summarizing my work and results and by discussing limitations and further research.

2 SPI theory

This chapter provides an overview of the SPI field as a background for my research question. First, the SPI field is introduced and its origin in CMM (Humphrey, 1989, Paulk et al., 1993) is presented. Then the special challenges that small and medium-sized organizations face when improving software processes are described. Finally, I explain my grounding in the field as part of a Danish stream of research on SPI. The chapter is concluded by stating my research questions.

2.1 The research field

The publication of the Capability Maturity Model (CMM) (Humphrey, 1989, Paulk et al., 1993) sparked a new interest in SPI and a field of both theory and practice of improving system development formed: Software Process Improvement (SPI). Now some 20 years later the CMM(I) suite (Paulk et al., 1993, The-CMMI-product-team, 2001, 2002) still dominates the field (Hansen et al., 2004a). The basic values and recommendations of the original are to a large extent unquestioned and unchanged (Ngwenyama and Nielsen, 2003, Hansen et al., 2004a), even though adaptations of the CMM (Sakamoto et al., 1996, Wilkie et al., 2005), alternative norms (e.g. BOOTSTRAP, see Kuvaja, 1999) and other ways of assessing (Fayad and Laitinen, 1997, Iversen et al., 1998a, Steel, 2004) have been widely discussed.

The research field of SPI is an applied academic field. The majority of the research forms a learning cycle between the theory prescribed by the research and the software industry testing the models in practice. Within this rather closed cycle, the dominant approach appears to be successful, but it is not clear that "it is widely appropriate or successful outside its natural habitat" (Hansen et al., 2004a). One example of a non-natural habitat seems to be small and medium-sized enterprises (SMEs). Evident in the field is an ongoing discourse on how to handle the special challenges SMEs face when adopting SPI technology (see section 2.3).

The field tends to be a prescriptive (or at least non-reflective) academic field that is overly focused on applied techniques in opposition to building defensible theory (Hansen et al., 2004a).

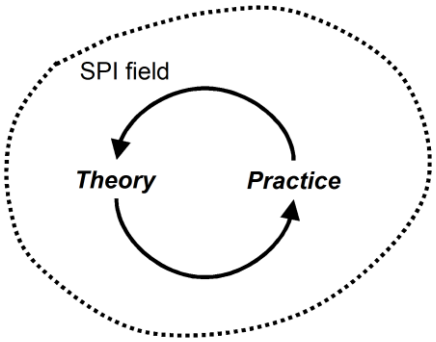

Figure 2.1 The learning cycle of the SPI field.

The prescribed approaches are mainly norm-driven but are supplemented by some problem-driven approaches (Hansen et al., 2004a, p. 460). Success stories from firms adopting CMM (e.g. Humphrey et al., 1991, Dion, 1992) are symbols in the field of the success of CMM. Hardly any failure stories have been reported even though a statistical survey from the Software Engineering Institute showed that 63% of 167 CMM organizations were at level 1 (the "stuck in first" phenomenon (Johnson and Brodman, 1996)) and only 11% were at level 3 and above (Herbsleb and Goldenson, 1996).

This main part of the SPI research field is supplemented by a more balanced, mainly Scandinavian, literature reporting both the difficulties and the advantages of adopting SPI technologies. This often takes the form of case studies involving some kind of theoretical framework (Hansen et al., 2004a, p. 464).

2.2 The origin of the field

As described above, the origins in CMM (Humphrey, 1989, Paulk et al., 1993) still dominate the field of SPI. Here I present important values and principles of CMM, mainly based on "Managing the Software Process" by Watts S. Humphrey (1989). SPI as a research field emerged with this work. It was based on collaboration between a group of researchers at the Software Engineering Institute (SEI) and a US Air Force project in search of ways to select capable software contractors. It resulted in the publication of the CMM (Paulk et al., 1993). The fact that the US Department of Defense utilized the maturity model to evaluate suppliers initiated an enormous interest in CMM both in the software industry and in the fields of information systems and software engineering research.

The CMM rests on the tradition of total quality management (TQM) (Deming, 1982) and inherits a set of values and assumptions about statistically controlled manufacturing processes and their optimization (Humphrey, 1989, p. 3). The core is that software production should follow defined processes to obtain a stable process under statistical control so that the outcome is predictable. The entire software task

is treated as a process that can be controlled, measured and improved. A process is defined as "that set of tasks that, when properly performed, produces the desired result" (Humphrey, 1989, p. 4). A better outcome of the process can be reached by improving the process itself. Improvement of the process is done on the basis of measurement according to a norm (the maturity model) and carried out through careful planning and preparation.

The CMM describes five levels of process maturity according to which organizations can be assessed and thus obtain guidance on where to start improvements. The levels range from ad-hoc software processes up to controlling the software processes to a degree where continuous and controlled improvement of the processes is possible. Each level prescribes best practices within a series of key process areas with which organizations should comply (Paulk et al., 1993, The-CMMI-product-team, 2001, 2002). The model serves in the SPI field as a norm for good manufacturing practices in software development and the norm-driven approach to improving practice serves as the convention for good process improvement.

Norm-driven SPI assembles a traditional learning cycle. It involves (Humphrey, 1989 p. 30):

- assessing and understanding the current software process according to a prescribed norm in order to decide what to improve (unfreeze)
- planning and implementing the changes (move), and
- sustaining the new processes through training and monitoring (refreezing).

In this learning cycle, the norm is the key since the goal is external certification.

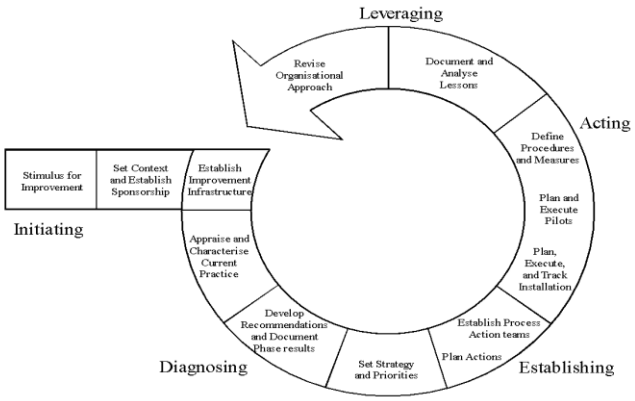

Figure 2.2 The IDEAL model (McFeeley, 1996) implementing the learning cycle of CMM

The IDEAL model (McFeeley, 1996) has become the de-facto standard for how to organize this learning cycle.

It is recommended that the improvement activity should be organized so that it is both centralized and separate from the core software production. SPI has to be driven from the top (Humphrey, 1989, p. 19). To carry the major changes through, leadership is needed and top management must set priorities and furnish the resources. Software organizations should form a separate software engineering process group (SEPG), staffed by dedicated full-time staff to drive the improvements as change agents (Humphrey, 1989, p. 289). The group initiates and sustains the changes as well as supporting normal operation. Software engineers should be involved in designing new procedures, as they are the most knowledgeable and ultimately everyone should be involved in the implemented improvements (Humphrey, 1989, p. 293 and p. 19).

The CMM norm has been subject to changes through continuous testing in the software industry, leading to improvements. The most important was when the CMMI was published (The-CMMI-product-team, 2001, 2002). On the one hand, CMM was from the start presented as one rather reasonable model offering software organizations an improvement path. Also organizations were encouraged to participate in the development of the model by testing adaptations and publishing the results. On the other hand, through the publication of the detailed norm and the development of the assessment industry the norm has become a de-facto standard, a one-size-fits-all by which organizations are assessed and certified.

2.3 Small and medium-sized enterprises

Paulk has argued that the adoption of CMM in small organizations "may be different in degree, but they are not different in kind" (Paulk, 1998) from those of other organizations. However it is widely recognized that SMEs face special challenges when trying to adopt SPI technology.

SMEs are highly sensitive to dynamic environments (Mathiassen and Vainio, 2007) and more vulnerable than larger enterprises. They lack both the resources to invest in improvements (Brouse and Buys, 1999, Kautz and Larsen, 2000) and SPI knowledge (Steel, 2004), and they find the SPI theory and the main approaches bureaucratic (Kelly and Culleton, 1999) and too costly (Villalon et al., 2002). Adding to this, an early study finds that CMM does not fit SMEs (Brodman and Johnson, 1994).

Since the European software industry especially is dominated by SMEs, much research has targeted this challenge. Some results recommend tailoring CMM to fit small organizations' needs (Batista and Figueiredo, 2000, Horvat et al., 2000, Kautz et al., 2000, Kautz and Thaysen, 2001, Casey and Richardson, 2004) while others evaluate CMM according to these needs (Wilkie et al., 2005). Yet others develop alternative approaches resting on the same principles as CMM but tailored to the resources and culture of smaller organizations (Kautz, 1998, Iversen et al., 1999, Kautz, 1999). Examples are; the 3P approach (Brouse and Buys, 1999), IMPACT (Scott et al., 2001), Software Process Matrix (Richardson, 2001, Richardson, 2002),

MESOPYME (Villalon et al., 2002) and COA (Grechenig and Zuser, 2004, Steel, 2004). Very little research has questioned the basic values of CMM, however.

The status of the discourse is addressed in a recent literature survey of reported case studies on SMEs adopting SPI technology. The survey shows that SMEs do adapt and utilize SPI technology in their improvement efforts: CMM (25%) (Paulk et al., 1993, The-CMMI-product-team, 2001, 2002), SEI models in all (51%), IDEAL (13%) (McFeeley, 1996) and the ISO standards (31%) (Hoyle, 2005). However SMEs rarely achieve formal certifications. The study by Pino, García and Piattini (2008) concludes that "it is indeed very difficult to successfully apply formal SPI programmes which use models such as for example CMM, to SMEs" (p. 253) and "we consider that these standards per se are not suitable" for small organizations (Pino et al., 2008, p. 248).

In summary, SMEs do practice SPI based on the dominant approaches, but they still do not succeed in the sense of certification.

2.4 My SPI research background

In this section I focus on my own research background in two major Danish research projects on SPI carried out between 1997 and 2006 (Mathiassen et al., 2002, Nielsen and Kautz, 2008). The research was organized as collaborative practice research with the emphasis on action research (McKay and Marshall, 2001, Mathiassen, 2002) following the tradition of Scandinavian IS research. The aims of the research were dual as the projects have both tried to improve practices in concrete organizations and to learn about practice in order to theorize based on experience. Most of the contributions from these research projects are either descriptive, with some prescriptive advice, or reflective.

The first project is reported in the book *Improving Software Organizations: From Principle to Practice* (Mathiassen et al., 2002). While the rhetoric of SPI says that assessing the capability of the organization and developing and implementing a strategy for improvement will lead to increased quality and productivity, the researchers' experience shows that it is not so easy and straightforward. They suggest five core SPI principles that must be adopted by organizations in order to succeed with SPI: (1) focus on problems, (2) emphasize knowledge creation, (3) encourage participation, (4) integrate leadership, and (5) plan for continuous improvement. "The five principles are a coherent philosophy of SPI" (Mathiassen et al., 2002, p. 20) developed through practice and based on values that differ from those of dominant SPI theories.

The second project was an offshoot of the first by focusing on knowledge management in SPI (their second principle) and is reported in the book *Beyond Conventional Software Process Improvement* (Nielsen and Kautz, 2008). The book goes beyond the project as it also contains theoretical reflections on SPI and reports from other research efforts done in parallel. The book brings nine contributions

organized in three parts: (1) "frameworks" focusing on central frameworks, e.g. CMM; (2) "techniques" focusing on more concrete knowledge-based techniques for SPI; and (3) two longitudinal "tales" of SPI spanning respectively 8 and 10 years. My research was part of this second project (see section 4.2.1.) and I contributed as chapters in the book revised versions of two of the papers which form the basis of this thesis (papers 3 and 4).

In general the Scandinavian SPI research has displayed a tendency to raise critical voices towards the dominating SPI approaches. I will here highlight three contributions that all characterize CMM through theoretical analysis, and point to inherent problems of this dominant SPI approach.

First, Ngwenyama and Nielsen (2003) investigate the assumptions about organizational culture embedded in the CMM models. They find contradictory sets of assumptions that could lead to significant problems in implementing SPI in organizations. In short, "the design ideal of CMM is the rational bureaucratic learning organization that is flexible" (Ngwenyama and Nielsen, 2003, p. 108). CMM is based on this rational ideal, but expresses allegiance with the developmental culture. The underlying rational culture makes CMM less effective as an approach to deal with the massive and deep changes of organizations that are prescribed by the model itself.

Second, Rose, Aaen and Nielsen (2008) outline CMM's underlying assumption platform and discuss the trouble with CMM. The underlying assumptions of CMM are: process orientation; hierarchical management – planning, monitoring, control; externally imposed generic process models; documentation, standardization and institutionalization; organizational progression to maturity; objective measurement, external verification and certification; and goal-directed change through rational analysis and learning. This forms a platform that was typical for large industrial production companies in the late industrial age. Analyzing the problems of applying CMM leads to the general observation that they often stem from applying an approach with a particular management philosophy that does not fit the target organization. Rose, Aaen and Nielsen conclude that CMM is narrowly applicable in organizations that share or can tolerate the underlying assumption platform. Since this kind of organization is decreasing in number in the information age, CMM may well be increasingly inappropriate.

Third, Aaen (2003) labels CMM "Blueprint SPI". "Plan-oriented and mainly concerned with the static, this method creates a *blueprint* of a future software process" (Aaen, 2003 p. 86). Blueprint SPI externalizes process knowledge, separates process design from use and structurate by melding process parts into wholes. This induces a high risk of confusing information publication with knowledge building, of seeing the process models as ends rather than means for improvement, and of underestimating the importance of tacit knowledge. Blueprint SPI tends to plan for the expected rather than the unexpected. This planning will

rarely be the best answer given the complexities and uncertainties of software projects.

In summary these authors criticize CMM for being too rational to deal with the complex and massive changes that it imposes on organizations, for resting on an old-fashioned managerial assumption platform not suited for modern organizations, and for being unable to plan for the unexpected that is a common aspect of software projects.

2.5 Research questions

My practice experience (see section 1.1) and my first study of the SPI research field (see section 2.1) resulted in doubts as to whether the theory of the SPI field actually expresses an appropriate understanding of the practice of the field. This doubt is supported in the three contributions cited above (Aaen, 2003, Ngwenyama and Nielsen, 2003, Rose et al., 2008) which are theoretically based. From this doubt I have phrased my research questions:

Research question 1: What is the problem with the dominant SPI theories' understanding of SPI practice?

Research question 2: What characterizes SPI practice?

Research question 3: What new theories explain this practice better?

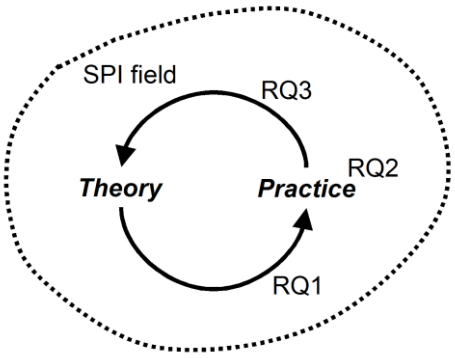

Figure 2.3 Mapping the research questions to the learning cycle of the SPI field.

Research question 1 (RQ1) addresses the understanding of SPI practice that underlies the dominant SPI theories and is expressed in their prescriptions. Research question 2 (RQ2) addresses the complex phenomenon that SPI practice is. The question focuses on SPI practice on its own terms by leaving the prior knowledge of the SPI field behind. Research question 3 (RQ3) aims at providing feedback to the research field of SPI by suggesting theories that better explain SPI practice.

3 Reference theories

This chapter presents my reference theories. First Ciborras' view on the adoption of technology as governed by drifting forces (Ciborra, 2002). This has been my main reference theory, helping me to interpret my findings from SPI practice. It will be supplemented with the theory of organizational improvisation (Cunha et al., 1999) used in the more focused case study on improvisation.

3.1 Drift theory

In the field of information systems, rational models about managing organizations and technology play a dominant role. When organizations strive to manage and control adoption of technology in accordance with these models, they most often experience that the adoption drifts away from the goals with unpredictable results. When organizations experience drifting, their perceived need for more and better control is reinforced. Ciborra denotes this a vicious cycle for organizations to be caught in .

The use of these rational management models constrains our understanding of the world and prevents us from seeing the full complexity of technology (Ciborra, 2002). Enforcing these simplistic geometrical models as understandings of the much more complex world is not in accordance with the world as experienced in the everyday life of agents, users, designers, and managers. This misuse of the management models Ciborra counts as a hidden or at least unrecognized crisis of the field of information systems.

The experienced drifting is due to forces like turbulent environments, complexity of the technology and the implementation process, side-effects, plain surprises, and users' resistance and creativity.

> Drifting can be looked at as the outcome of two intertwined processes. One is given by the openness of the technology, its plasticity in response to the re-inventions carried out by users and specialists, who gradually learn to discover and exploit features, affordances, and

> potentials of systems. On the other hand, there is the sheer unfolding of the actors' being in the work flow and the continuous stream of interventions, tinkering, and improvisations that color perceptions of the entire system life cycle. (Ciborra, 2002, p. 87).

When this happens, usage, maintenance and redevelopment, and improvement of technology take place simultaneously. It can involve acts of many kinds ranging from sabotage, to passive resistance, to learning-by-doing and to micro discoveries and radical shifts (Ciborra, 2002, p. 89).

The results of drifting can be very beneficial for organizations, because humans are bounded in their technological imagination by, among other things, the specific formative contexts as described in (Ciborra and Lanzara, 1994) and thus have limited innovative capabilities. Coincidence and breakdowns followed by human coping can spark technology drifting, that result in unthinkable innovative outcomes. When technology adoptions drift away from the plans, humans respond by reinventing the technology through improvisations, tinkering, bricolage, and hacking.

To benefit from this potential innovative power, organizations need to change their thinking and practices from control to drift. Such a move will allow organizations to support human innovation instead of controlling plans and to facilitate cultivating and hosting of technology instead of trying to plan or design it.

Ciborra picked CMM as one of his examples of an inappropriate and limiting model (Ciborra, 2002, p. 19). He provides new concepts from the drift theory that allow an understanding of SPI practice more in line with the complex modern world.

How this drift theory is used in my research is described in more detail in paper 2 (Tjornehoj and Mathiassen, 2008a, see section 5.3.).

3.2 Organizational improvisation

Improvisation has been suggested as a way of coping when time pressures hinder rational planning, decision processes, and knowledge creation (Cunha et al., 1999). Traditionally improvisation is seen as the deviation from the norm of rational decision-making. As uncertainty, complexity, and environmental dynamics increase as conditions for organizations, their ability to improvise becomes more important (Chelariu et al., 2002).

The defining characteristic of organizational improvisation is convergence between planning and execution of actions. Improvisation is triggered when something unexpected occur that demands immediate action.

> … thus improvisation arises when both (1) a demand for (a) speed and (b) action, and (2) an unexpected (and unplanned for) occurrence are perceived by the organization. (Cunha et al., 1999).

Cunha et al. (1999) also highlight that improvisation is deliberate, extemporaneous, and occurs during action, drawing on "…available material, cognitive, affective and social resources" (Cunha et al., 1999). This last characteristic connects improvisation to bricolage by emphasizing that planning and action need to take place within the limits of available resources and knowledge to be called improvisation.

Important conditions for the ability to improvise in organizations are an experimental culture, a (minimal) control structure, and a low procedural memory or small number of routines (Cunha et al., 1999). An experimental culture values action and experimentation when trying to understand and deal with reality. A control structure is required for focusing, coordinating, and keeping the necessary feeling of urgency, but it should be minimal so as not to restrict the participants. Procedural memory is the amount of routine knowledge that the organization possesses. If the procedural memory is low, it leaves more room for improvisation since more events are unplanned. On the other hand, a high procedural memory perceived as adaptable knowledge instead of unbreakable rules will also enhance improvisation.

Improvisation can have both positive and negative results. Possible positive outcomes include motivation, flexibility, increased ability to improvise, gaining new knowledge, and new routines and practices. Among the negative results is inappropriate learning biased by actual circumstances, opportunity traps by not acquiring new knowledge, over-amplifying emergent events and addictiveness to improvisation thereby under-utilizing existing knowledge and skills. Employees also face increased anxiety and uncertainty (Cunha et al., 1999).

According to Ciborra, the modern world with its increasing uncertainty, complexity, and environmental dynamics causes drifting technologies. He thus suggests organizations should 'host technologies' by embracing new technology as a guest, leaving room for improvisation and mutual adaptation instead of rationally planned adoption. The theory of organizational improvisation (Cunha et al., 1999) describes parts of this. Improvisation can play a role in the adoption of technology especially in organizations that lack resources and are vulnerable towards dynamic environments such as SMEs.

How the concept of improvisation is used to understand the case study is described in further detail in paper 3. (see section 5.4.).

4 Research design

When designing research projects the research question and objectives determine which research approaches would be appropriate to secure valid and relevant results. However, research design is also shaped by given opportunities and practical issues especially regarding the research organization. In this chapter I first present the IS research framework (Braa and Vidgen, 1999) to argue my choice of research approaches before I introduce these approaches. Second, I describe my research project in more detail.

4.1 Research approach

In this section I argue my choice of research approach based on the IS research framework (Braa and Vidgen, 1999) and present my approaches – interpretative longitudinal case studies (Pettigrew, 1990, Walsham, 1993, 1995, 2006).

4.1.1 The IS research framework

The IS research framework outlined by Braa and Vidgen (1999) presents the variety of research approaches that is utilized in IS research as a triangle.

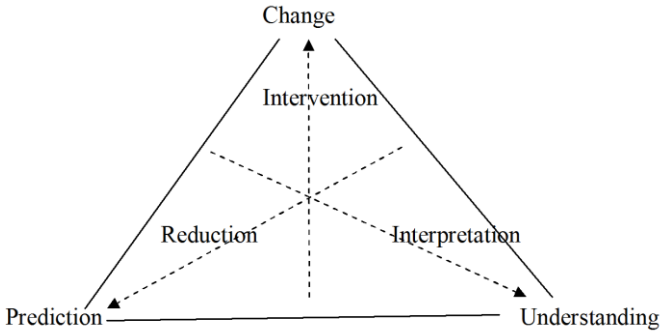

Figure 4.1 The IS research framework from Braa and Vidgen (1999)

Each point represents a different research outcome. The main approaches to achieve these goals are illustrated as the arrows; a reductionist approach to predict the future, intervention into practice to bring about change, and interpretations to understand the world.

Braa and Vidgen mapped the well known research methods to this framework by categorizing action research, field experiment and soft cases as pure methods aligned to intervention, reduction and interpretation, respectively, while quasi-experiments, hard case studies and action case represent some of many hybrids.

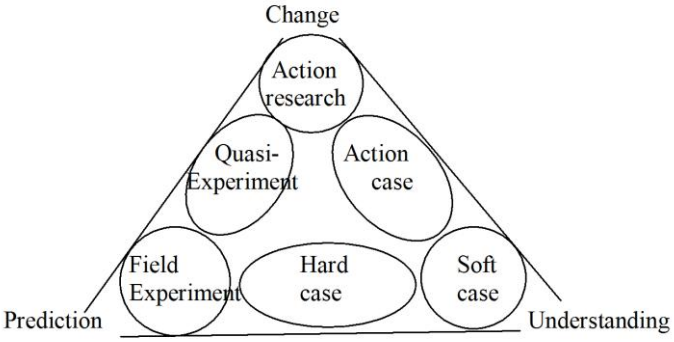

Figure 4.2 Research methods mapped onto the IS research framework (Braa and Vidgen, 1999)

The triangle pinpoints the contradictions (or dilemmas) that has to be dealt with in the 'dilemmatic' (McGrath, 1982) process of designing research projects. For example, it is difficult to mix. striving to reach a rich and deep understanding of complex situations with prediction by cause-effect relationships, since this involves reduction of complexity. Also deep involvement in an actual situation in order to bring about changes does not fit the idea of being an observer collecting rich data for interpreting the situation. The hybrid methods represent design trade-offs taking two of the points into account, but neglecting the third. The center of the triangle represents a desired but unlikely super method since the trade-offs cannot be resolved altogether (McGrath, 1982).

The research field of SPI is traditionally dominated by research in the left two points of the triangle as seen in the many descriptive and prescriptive contributions reporting on experimenting with the prescribed methods of SPI (Hansen et al., 2004a, section 2.1).

I have placed the main part of my research in the lower right point of the triangle. I have conducted a soft case study collecting and interpreting rich data from the history of SPI practice of a firm. This allows for gaining deep and profound understanding of complex realities, and fits my research question 2 and 3 well. It is also in line with my second research goal of going beyond the most common literature of the SPI field by providing reflective and independent research.

Whether I reach my first research goal, of providing research results relevant for the software industry to improve their system development practice, is mostly dependent on having a relevant research object and topics and on the chosen publication form. My first research question is mostly served by my literature review (See section 5.2).

4.1.2 Interpretive longitudinal case studies

My main research approach has been longitudinal case studies as described by Pettigrew in "Longitudinal Field Research on Change: Theory and Practice" (Pettigrew, 1990) and it has been interpretive in nature as described by Walsham in his book and two papers on interpretive research in the IS field (Walsham, 1993, 1995, 2006). Below I show how I have combined the two approaches to supplement each other in one single approach.

Pettigrew characterizes change as multifaceted and shaped by power, chance, opportunism, accident as well as design, negotiation and planning. He states that "sound and practically useful research on change should explore the contexts, content and process of change together with their interconnections through time" (Pettigrew, 1990). If we want to understand change we need to avoid the traditional simplistic view of change as planned, linear and rational. This can be done by applying contextualism and a processual view in a holistic and dynamic analysis drawing from both vertical (higher and lower levels of analysis) and horizontal (historical, present and future time) levels of analysis and from the interconnections between them over time (Pettigrew, 1990).

The core of interpretive research can be captured through the underlying worldview. "Interpretive methods of research start from the position that our knowledge of reality, including the domain of human action, is a social construction by human actors" and thus "theories concerning reality are ways of making sense of the world" (Walsham, 2006 p. 320). This implies that the researcher never can take a neutral stance as he himself interprets the data that actually result from other humans' interpretations of reality. Interpretive research enables the researcher to reach in-depth knowledge and understanding of complex social processes for the benefit of future processes.

When done well the research is iterative and characterized by periods of expanding complexity through collection of more data and open analysis and of periods of simplification through use of theory and data reduction. Different kinds of output will emerge that are suited for different audiences: analytical chronology, diagnostic, and interpretive or theoretical cases and eventually if appropriate; meta level analysis over multi-case studies (Pettigrew, 1990). It is a both timely and resource-demanding kind of research that takes good social skills and involves a lot of practical issues to solve (Pettigrew, 1990, Walsham, 1995, 2006).

When organizing the fieldwork, choice of research site and considerations of time and of data collection should be done carefully. It has to allow for triangulated collection of data, which is processual, comparative, pluralist, historical, and contextual (Pettigrew, 1990). Walsham introduces the notion of 'thick description' (1995 p. 75) from the anthropological tradition as a way to handle the resulting wealth of rich data.

Walsham (1993) underlines that "good theory and insightful analysis" is the key in the work. Theory can be used in different ways: (1) as an initial guide to design and data collection, (2) as part of an iterative process of data collection and analysis and (3) as a product of the research (Walsham, 1995 p. 76). Theory should be used carefully in an inspiring and flexible manner that allows for discarding it altogether, even if it has played an important role in the work.

Four kinds of generalizations are possible from interpretive research: (1) development of concepts, (2) generation of theory, (3) drawing of specific implications, and (4) contribution of rich insight (Walsham, 1995 p. 79).

This kind of research cannot be measured by the traditional scientific quality criteria, since it is grounded in a totally different worldview. Walsham suggests using the rather simpler criteria used in ethnography when evaluating if the research is convincing and sound: authenticity, plausibility and criticality (Golden-Biddle and Locke, 1993).

4.2 Research project

4.2.1 Research organization

My research has been framed by the Danish collaborative practice research project, Software Processes and Knowledge (SPK). Collaborative practice research (CPR) projects (Mathiassen, 2002) aim to resolve the tension between the points of the IS research framework (Braa and Vidgen, 1999) by balancing relevance and rigor in one research project through close collaboration with practitioners and a flexible multidisciplinary approach (Mingers, 2001).

The constituting research approach of CPR is action research (Checkland, 1991, McKay and Marshall, 2001) with its outset in the practitioners' view of their practice. At the same time the research interest will aim for more general knowledge. These contradicting goals of CPR are negotiated by the establishment of a sound relationship between researchers and practitioners to guarantee relevance of research and at the same time to structure and manage the research to produce rigorous results.

The ultimate goal of the SPK research project was to improve software development practice. The SPK project involved four Danish software organizations and 10 researchers from three research institutions. Together they

formed an organization serving both interests, as recommended by Mathiassen (2002). The primary action in practice was organized in four local research groups. The interaction and knowledge sharing between these local groups took place half-yearly as a plenary session and the research interests were supported and secured through a researchers' forum that met regularly.

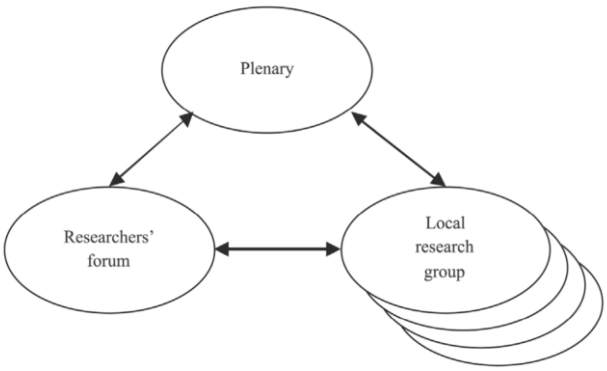

Figure 4.3 The organisation of the SPK project in line with CPR (Mathiassen, 2002)

The project ran for a little more than three years (2002–2006) and resulted in a portfolio of very diverse research results published in many different research outlets. To sum up the project a book with some of the results was published, Beyond Conventional Software Process Improvement (Nielsen and Kautz, 2008). This form of reporting traditionally suits practitioners better than journal articles. Revised versions of both paper 3 (see section 5.4) and paper 4 (Nielsen and Tjørnehøj, 2009, section 5.5) are published in the book.

The SPK project primarily supported the fieldwork of my research, investigating SPI in the smallest of the participating organizations. It also placed me in a community of senior researchers that supported my learning and provided a broader view on SPI through the work of the research plenary. The conglomerate of practitioner interests and problems, researchers and research approaches, specific research questions and theory applied has informed my work in a very useful manner.

4.2.2 Emergent research design

My research design has been emergent in the way CPR allows for emergent research design (Mathiassen, 2002). Mainly three events have changed my plans and influenced the final research project dramatically. First, the fact that I was invited to participate in the SPK project provided an unexpected and helpful research organization and gave me access to the case-study firm. Second, when I had planned action research in the firm, they unexpectedly had to withdraw, since they experienced a financial crisis. Third, an interview with the SPI manager of the firm on management commitment suddenly turned into an interesting discussion on

past, present and future SPI initiatives. This interview inspired me to change track towards the resulting case study. These events all contributed to shaping the resulting research design but have not influenced my research goals and interest in understanding SPI practice.

Except for these introductory remarks I will not dwell on the history of emergence of the research design nor will I describe all the plans that did not come to fruition. I will instead describe the resulting research design of my research project in some detail.

4.2.3 Overview

The overall design is a longitudinal case study in a small Danish software firm, *SmallSoft*. The case study was interpretive in nature. A small action research study in which I participated is reported here as an integral part of the case study. As described above, my research was framed by the SPK project.

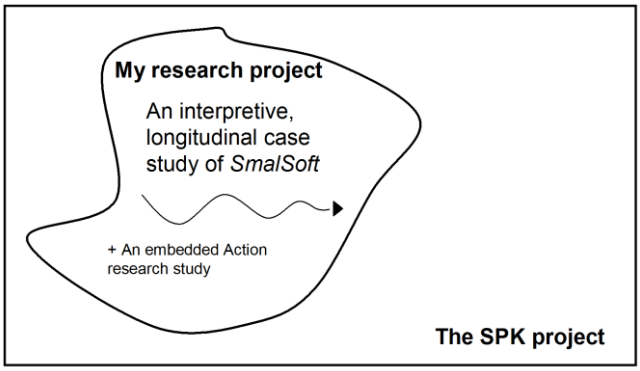

Figure 4.4 Organizational map of my research project.

My research has been carried out over a period of six years (which also involved half-time teaching). The first three and a half years were spent on literature studies and fieldwork, while the last two and a half has been focused on completing the papers for publication and writing the summary.

The first activity was a literature review of the research field of SPI. We collected references from well-known journals and published proceedings of conferences in the field to form a database on SPI literature. We categorized the contributions according to a framework as prescriptive, descriptive or reflective and could thus characterize the shape of the SPI research field. The study is reported in paper 1 (Hansen et al., 2004a, section 5.2). This was followed by the action research intervention in *SmallSoft*. The firm wished to change the organization of its SPI effort radically, and the intervention aimed at supporting management decisions on this reorganization.

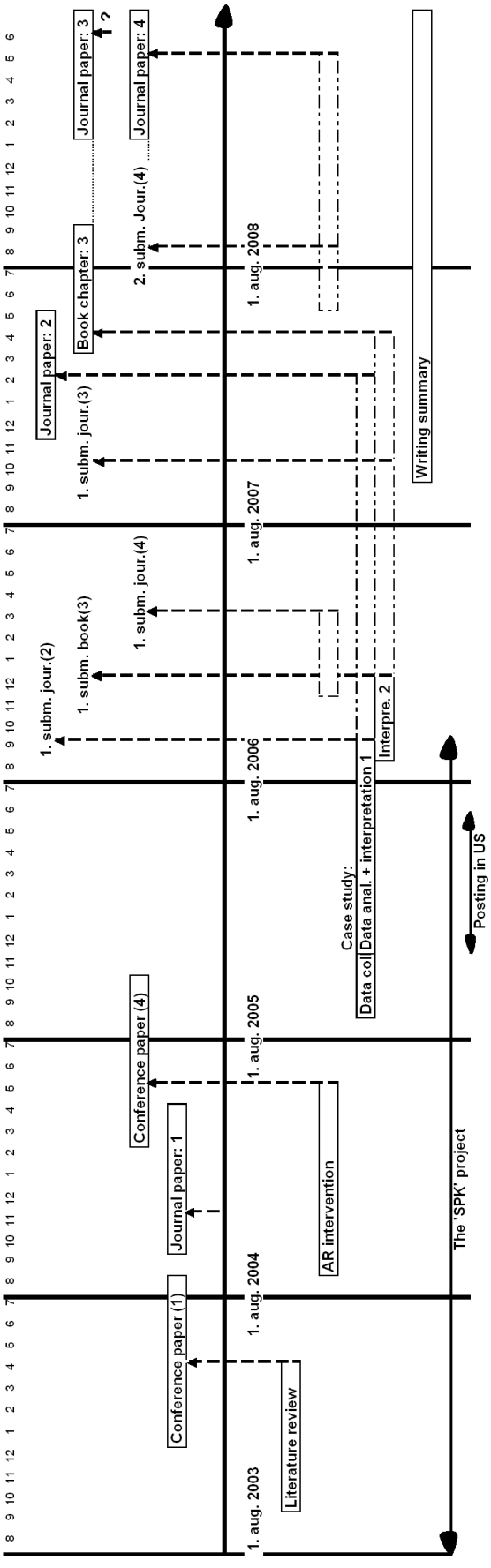

Figure 4.5 Historical map of my PhD work showing activities and papers from 08/01/03 to 08/01/08 and beyond.

Based on our previous knowledge of the firm and on collected network data, we conducted a social network analysis of the SPI communication networks of the firm. Mappings of the networks facilitated management negotiations. We report from this research in paper 4 (Nielsen and Tjørnehøj, 2009, section 5.5) We found the results rather promising, but for practical reasons the intervention in the firm was terminated. In this thesis, the study mainly serves as part of the case study.

The next research effort was inspired by an exciting discussion with the SPI manager of *SmallSoft* on the history of their successful and unsuccessful SPI efforts. In collaboration with a colleague I decided to carry out an interpretive longitudinal case study of SPI practice in *SmallSoft*. My colleague knew *SmallSoft* well through participation in their former SPI efforts, and I had actually been employed there for a short time. We had very good access to the firm and to data from a long historical period. The study was initiated by a three-month short data collection period, followed by a longer period of data analysis and interpretation. The first analysis was mainly conducted on a posting in the USA. We report from this work in paper 2 (Tjornehoj and Mathiassen, 2008a, section 5.3).

A second analysis of the case, focused on organizational improvisation is reported in paper 3 (see section 5.4).

I have been engaged in writing-up the PhD summary on and of from November 2007.

4.2.4 The case study

When designing a case study, there are a lot of practical issues that have to be handled well to result in good quality research. The practical setting and the resolution of the issues do count when evaluating this kind of research and are thus expected to be reported in some detail. In this section I describe how I have handled the focus areas to which Pettigrew (1990) and Walsham (1993, 1995, 2006) draw attention (see also section 4.1.2).

Choice of research site

Being part of the SPK project led to an easy choice of research site. The smallest participating firm was rather typical for the Danish software industry. It was a small (<100) and relatively young (<30 years) organization with a niche production. The employees were a mix of software-educated and other specialists. Being typical it fits into my first goal of providing relevant knowledge for the software industry. As laid out in section 2.3, being small also means that they are most likely to encounter some of the "typical" problems of SMEs adopting SPI. In addition to this the firm, was known as trustworthy among the researchers through collaboration in different learning and networking activities. As part of the CPR project we were welcomed by the firm, and granted access with no or very few limitations.

When my research design found its final form, the firm had just experienced a difficult time and the managers were themselves reflecting on the development of the firm; how new practices had emerged, under what circumstances and by what means, thinking that they could learn from it. We took this idea to the level of proper research by designing a longitudinal interpretive case study to understand the changes in the practice of the firm over the years as more than rationally planned and implemented events.

Considerations of time

Matters of time in this research project were most often settled by practical issues. When the firm involved me in their reflection on past and present improvement efforts, they opened the opportunity for this research project. We then decided to collect as much historical data on the adoption of SPI technology in the firm as we could. In the mind of the SPI manager, this history of improvement started with the design and implementation of their QA system back in 1996. Tales of the improvement culture of the firm reached even further back. Our data collection stopped in December 2005, since I went abroad as part of my PhD study. Thus the timespan of the researched SPI practice is from the introduction of the QA system in 1996 until the new matrix SPI organization was implemented in 2005 just before I left.

We were aware that this is only a glimpse into a still ongoing change process. We found the beginning time-limit appropriate since the written material goes back to the introduction of the QA system, and the ending time-limit also, since all

participants (firm and researchers) had other responsibilities by the end of 2005. The period is sufficiently long because it covered the lifespan of several and very different SPI efforts.

Fieldwork and data collection

Both researchers involved have had substantial though periodic contact with the firm during a period of ten years in different roles; consultant, supervisor, employee, and researcher. This involvement in the past combined with the more recent collaboration in the SPK project accounts for the fieldwork of the study. It allowed us to collect substantial amounts of different kinds of data from a diversity of sources as sketched in Table 1 below.

Description	Source	Type	Dated	Covering
A detailed internal report documenting SPI assessment and planning of SPI activities. Carried out by 3 key employees under supervision of an SPI expert, as part of an official SPI education.	Firm archive	Documentary	2001	1996 – Spring 2001
Documents from the SPI efforts of the firm (agendas, memos, reports, quality assurance documentations etc.)	Firm archive	Documentary	2001– 2005	April 2001 – Nov. 2005
Reports from students projects in the firm. Subjects within SPI.	AAU report archive	Research/ consulting by students	2002	Spring 2002 and Fall 2002
Research notes (written debriefings from research interviews, personal notes and dairy pages)	Researchers archive (2)	Observations	2003	Spring 2003
Email correspondence between researchers and firm organizing the SPI effort in the local research group.	Researchers archive	Documentary	2003– 2004	2003 – 2004
Recorded meetings in the firm: quality assurance meeting (March 2003), management meeting on SPI (researchers participating actively) (March 2005) and kick-off meeting for new SPI organization (August 2005)	Research archive (1) + own collection (2)	Documentary	2003 – 2005	Spring 2003 – Fall 2005
Interview with the SPI manager and 3 key employees. The last interviews guided by a historical mapping of SPI efforts in the firm.	Own interviews	In-depth interviews	2004 + Nov. 2005	1996 – Nov. 2005
Social networks mapping – "questionnaire" and results	Own research	Action research	Oct. 2004	2004
Interviews with 2 researchers from the local research group (among others commenting on the written material)	Own interviews	In-depth interviews	Nov. 2005	2003 – 2005

Table 1 The collection of data available in the case study

The data sources are both internal and external and represent a range of different actors; management, employees, students and researchers. Most time periods are covered by more than one source (and types of sources). The types of data are documentary, observations (own and others), interviews and previous research results. As the table shows, we had access to pluralistic, triangulated, historical and contextual data. To this material from sources outside ourselves, we can add substantial personal knowledge of the firm from being both insider (former employee) and outsider (consultant etc.).

One drawback is that only the documentary material is historical, while most interviews, eyewitness testimonials and own experience from the early part of the period is retrospective. However the only (not desirable) alternative was to discard the first time period of the case study all together.

Data Analysis and Interpretation

Having collected the first historical and retrospective data, we made a historical map of SPI events in *SmallSoft* during the period (an analytical chronology). This provided us with an overview and the chronology served as interview guide when interviewing the key employees of the firm.

To handle the time aspect we introduced the theory of encounters and episodes (Newman and Robey, 1992, Cho et al., 2008). We identified encounters and episodes that either the interviewee or we found to be important. We did so iteratively, describing the suggested encounters and episodes in more and more detail based on systematical data analysis. We focused on activities, events and actors within and outside the firm that influenced the adoption of SPI technologies. Through this process we tested the candidates for episodes and some were confirmed while others were modified or replaced.

We then applied Actor Network Theory (ANT) (Callon, 1986, Latour, 1987, Walsham, 1997) allowing us to identify and explore actants of the events and episodes and their interests in and influence on the adoption. Since ANT does not inherit any prior hypothesis or explanations theories, applying ANT helped us to avoid the "traditional simplistic view of change" that Pettigrew (1990) refers to, by allowing for shifts in levels of analysis and in focus as appropriate.

Together these two orthogonal analyses serve as the basic data analysis of my research. The interpretations of the case take their outset in the result of this basic analysis.

In the first interpretation we turned to the concepts of control and drift by Ciborra (2002) to explain and give meaning to the case. This was inspired by two findings from the basic data analysis. First, we found a wave-like pattern of encounters of SPI efforts that were experienced as successful, interesting and promising by those involved, but were followed by episodes of eroding results and fading energy.

Second, we found important and sustainable improvements that were grown from the grassroots. The resulting account of how the adoption of SPI technology was shaped between managerial control and drift was written up as a thick description. For further details see paper 2 (Tjornehoj and Mathiassen, 2008a).

This led to a second and supplementary interpretation of the case. We narrowed our focus to how organizational improvisation had been part of the SPI technology adoption. This interpretation took place within the understanding gained through the first interpretation and was based on the same basic analysis. Yet the research objective, the primary level of analysis and the analytical framework, was different and thus led to new insights from the same case. For the purpose of this interpretation we adopted the framework of organizational improvisation by Cunha (1999). The results are reported in paper 3 (see section 5.4.).

Writing up
When we wrote the first paper (Tjornehoj and Mathiassen, 2008a, section 5.3), we were advised by the reviewers not to report the ANT analysis, due to the overwhelming amount of theory that blurred the authenticity[2]. The ANT analysis had however served as an important scaffolding (Walsham, 1995, p. 76) for the results.

The results include generalizations of three out of the four kinds that interpretive case research allows (Walsham, 1995, p. 79):

- Generating of theory: e.g. the suggestion of "negotiating SPI between control and drift" as a development of the theory of "from control to drift" by Ciborra (2002) and the finding that the theory of organizational improvisation (Cunha et al., 1999) could not explain conflicting improvisations in the firm at different levels and serving different interests led us to suggest the concepts of micro and macro improvisations.
- Drawing of specific implications: e.g. the suggestions for managers when adopting SPI technology (both papers 2 and 3) is of this kind.
- The contribution of rich insight that stems from the detailed data analysis: e.g. the case descriptions of these papers.

4.2.5 An embedded action research intervention

The action research intervention serves as the first part of this case-study as we started data collection and studying of the SPI efforts of *SmallSoft* through the intervention. The result of the action research intervention is also reported on its own as paper 4 (Nielsen and Tjørnehøj, 2009, section 5.5).

[2] One of the quality critera for interpretive research from Golden-Biddle and Locke. The others being plausibility and criticality.

The initiating problem in *SmallSoft* for the intervention was that a centralized SPI organisation had failed. The SPI manager characterized the organization as "a complete failure" and looked for a completely different and more suitable way to organize and achieve the desired improvements.

We supplemented the data already collected as part of the SPK project with further inquiry, to reach a rather detailed understanding of the situation. Based on this we suggested and initiated an action research intervention inspired by social network analysis theory, in line with the procedure suggested by Cross and Parker (Cross and Parker, 2004). The intervention involved designing a graphical questionnaire to collect data of the SPI communication and knowledge networks of *SmallSoft*. The data was loaded into a tool for social networks analysis, NetDraw[3]. We used this tool for the iterative analysis of the network data looking for patterns that could reject or confirm our working hypotheses about *SmallSoft's* SPI activities. We validated the findings through the managers of the firm. The results were presented to management and led to a rather detailed discussion of the network problems and (a new) SPI organization in *SmallSoft*.

[3] NetDraw is available at www.analytictech.com/Netdraw/netdraw.htm.

5 Research Papers

In this chapter I present my research papers briefly. First I provide an overview and describe the different roles that each paper has played in my PhD work. Secondly, I present more detail on each of the papers.

5.1 Overview

Table 2 provides an overview giving the titles, authors, place of publication and research approaches of the papers.

#	Title	Authors	Publication	Approach
1	Prescription, description, reflection: the shape of the software process improvement field	Bo Hansen Jeremy Rose Gitte Tjørnehøj	Published in *International Journal of Information Management*, vol. 24 no. 6, 2005	Literature review
2	Between control and drift: negotiating improvement in a small software firm	Gitte Tjornehoj Lars Mathiassen	Published in *Information, Technology and People*, Vol. 21, 2008	Longitudinal interpretive case study
3	Improvisation during Process-Technology Adoption: A Longitudinal Study of a Software Firm	Gitte Tjornehoj Lars Mathiassen	Submitted to *Journal of Information Technology* 2007[4]. Revised manuscript submitted March 2009. A previous version published as a chapter in *Beyond Conventional Software Process Improvement* (Nielsen and Kautz, 2008)	Longitudinal interpretive case study
4	Social Networks in Software Process Improvement	Peter A. Nielsen Gitte Tjørnehøj	Published in *Software Process Improvement and Practice* 2009	Action research (CPR)

Table 2 The four papers that form the basis of this thesis.

Earlier or revised versions of some of the papers have been published; papers 1 and 4 in conference proceedings (Hansen et al., 2004b, Nielsen and Tjørnehøj, 2005),

[4] Administrative problems have delayed the reviewing process.

and papers 3 and 4 as chapters in the book, *Beyond Conventional Software Process Improvement* (Nielsen and Kautz, 2008).

The papers and the work that led to them have played very different roles in my learning process towards this PhD thesis. The literature study served as an introduction to the models, methods and practice understanding of SPI and gave me an opportunity to test my pre-conceptions of the field. It also laid the ground for my choice of research approach by establishing that reflections, deeper insights and independent studies are rather rare in the field of SPI.

The action research intervention gave me a chance to try out action research and to get back in touch with the case firm. The understanding of their SPI knowledge and communication networks gained from this intervention was very valuable in the further study of the firm.

The longitudinal interpretive case study is the core of my PhD. Having the opportunity to study 10 years of SPI efforts in *SmallSoft* in detail was invaluable. It allowed me to build an understanding from the rich data through interpretation of the case with drift theory and it provided insights and results of the real world. By this I mean that the full complexity of SPI practice has been taken into account and is not abstracted away. The combination of realism of the study and the theory that allows for this realism was very useful when building a profound understanding of SPI practice as drifting.

Based on my new understanding of SPI practice as drifting, the second interpretation of the case was the first investigation into when, how and why drift happens. In this study we looked at improvisation. In the paper we formulate recommendations for management on how to facilitate valuable improvisation. This is my first attempt to target the results at practitioners, and for my profound understanding to result in advice for practitioners.

The learning reflected in these papers, in the light of my research questions, has resulted in a new and documented understanding of SPI practice and its implications for the SPI field. This is discussed in chapter 6.

5.2 The literature review

In this section, I summarize the contribution of paper 1 (Hansen et al., 2004a).

Purpose:
The purpose of this study was to gain an overview of the research field of SPI, so that we could characterize it as a background for further studies in the field.

Approach:
We reviewed 322 representative research papers published in IS journals or proceedings of academic conferences. We found the references through an iterative

search process (for details, see paper 2). To qualify for the review, publications had to name SPI in the title, abstract or keywords. We reviewed the papers to gain an overview of the types and topics of the contributions and gathered the references in a database.

We developed a simple framework inspired by the evolution of an applied field of research. We would expect such a field to display a balanced cycle in which: (1) theoretically derived prescriptions are carried out in practice, (2) the resulting experiences are described to generate understandings that (3) again are reflected upon in order to form theory that can lead to better prescriptions. We could thus categorize the contributions as prescriptive, descriptive or reflective and through this characterize the SPI field of research. We described and summarized the different kinds of contributions and did a few simple calculations.

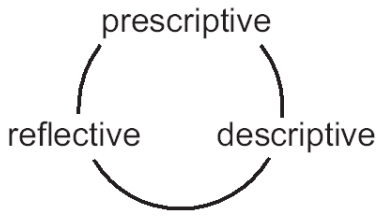

Figure 5.1 The simple framework used in the literature review

Findings and results:
We found that the applied field of SPI was biased towards prescriptive contributions and was dominated by one approach, CMM (72% of all contributions referred to CMM either in the title, abstract or keywords)

In the paper we address two issues in the field. First, we found an evident closed single-loop learning cycle formed by the SEI refinement of the CMM and the industry implementing the model and experimenting to learn how best to implement it. Success stories play a major role in the field, while failures are unreported, except in some Scandinavian research and between the lines in SEI's own figures. We question the success of CMM across environments and cultures. Second, we found the severe bias of the SPI field towards prescriptive contributions inappropriate even though prescriptions may be inherent in an applied field of research. As a result of this study, we suggested rebalancing the field by more independent and theoretical informed research focusing more broadly on the improvements of processes across the software industry.

Contribution:
The contribution is a critical literature review of the research field of SPI arguing that the field is dominated by CMM and lacks independent and reflective research.

This literature review was conducted in 2004. It is my perception that the SPI research field may have altered somewhat in shape since then. At least, the contributions of the SPK research project (Nielsen and Kautz, 2008) add to the pool of independent and reflective research. Through my work in writing up the thesis I have done some further literature searches and have found references that would also have raised the number of publications addressing independent reflective research, for example, on SPI agility (Börjesson and Mathiassen, 2005, Aaen et al., 2007, Allison and Merali, 2007).

5.3 The case study interpretation 1 – Between control and drift

In this section, I summarize the contribution of paper 2 (Tjornehoj and Mathiassen, 2008a).

Purpose:
The literature in the SPI field offers a number of studies on small organizations adopting SPI, but very few results on how such initiatives evolve over time. The purpose of this study was to investigate how adoption of SPI technology was shaped over a 10-year period in the small Danish software firm *SmallSoft*. Against this backdrop we try to answer the research question: "How can small software firms manage the adoption of SPI technology?".

Approach:
The investigation was based on an interpretive longitudinal case study of the improvements efforts in *SmallSoft* over 10 years (1996–2005). The data collected were diverse and from many sources, both internal and external to the firm. To some extent they were either retrospective (as interviews) or historical (from archives) and were combined with detailed knowledge of the firm and its history from collaborations by researchers with the firm over the years. We structured the study by focusing on encounters that impacted the improvement efforts, engineering or management practice of the firm. The encounters were chosen through a truly iterative data analysis process of reading and rereading the data. For further detail, see section 4.2.4. When interpreting the case we worked through the encounters again, writing the story of how SPI adoption in *SmallSoft* was shaped between managerial control and drifting forces such as improvisation.

Findings and results:
We found that the improvement effort in *SmallSoft* was fluctuating and shaped between managements attempt to control SPI technology adoption and events that caused the process to drift in unpredictable directions. This is described in a detailed process analysis in the paper.

Based on these findings we suggest that managers of small firms should remain flexible and constantly negotiate technology adoption practices between control and

drift, creating momentum and direction according to firm goals through attempts to control, while at the same time exploiting backtalk options and innovations from drifting forces inside and outside the firm.

As a theoretical result we recommend substituting the "from control to drift" perspective on organizational adoption of complex technologies like SPI with a "negotiating control and drift" perspective.

Contribution:
The paper contributes to the SPI literature by providing rich insights through a detailed and longitudinal case description of a SPI effort and by showing the usefulness of an alternative conceptual framework for understanding and describing this kind of practice. It contributes to the literature on organizational adoption of technology by suggesting an alternation of the concept "from drift to control".

5.4 The case study interpretation 2 – Organizational improvisation

In this section, I summarize the contribution of paper 3 (Tjornehoj and Mathiassen, submitted 2007)

Purpose:
Small software firms experience problems when adopting SPI technology. They lack resources and knowledge, the dominant SPI approaches fit poorly with their needs and they are highly sensitive to dynamic environments. Often improvisation is promoted as the means to resolve contradictions between pressure towards innovation and lack of resources. The purpose of the study is to investigate the role of improvisation in the adoption of SPI technology over a 10-year period in a small firm. We have tried to answer the question: "Why, when and how does improvisation shape the adoption of process technology in a small software firm?".

Approach:
This study builds on the same data collection and basic data analysis as paper 2 (see section 5.3). Also the analysis of improvisation in the case evolves within the understanding generated through the first analysis. In this second analysis we worked through the encounters and data once again, focusing on the role of improvisation in the case – describing when, how and why it happened.

Findings and results:
We found that *SmallSoft* was constantly improvising to meet unexpected events at all levels of the organization during the 10-year period. The firm's culture was experimental, with a low level of procedural memory, leaving room for much improvisation. The improvisations were of many types, degrees and on all levels and with very varying outcomes. We found that the improvising culture of the firm was a great strength in a turbulent environment. That is, improvisations addressed

appropriate challenges and were supported and coordinated to benefit the firm. However, improvising when there is no need because the events could have been planned for and because there is no real time pressure can jeopardize efficient production.

Based on these findings, we advise managers of small firms how they can exploit improvisation in the adoption of complex technologies by facilitating an appropriate improvisational culture.

Contribution:
Organizational improvisation is rather unexplored in the adoption of SPI technology. This study adds important empirical insights to the field from a longitudinal study. We identified two different levels of improvisations interacting, often uncoordinated and sometimes in contradiction. This understanding of levels of improvisation adds to the theory of organizational improvisations. We advise managers on how to facilitate an appropriate improvisational culture to ease the adoption of SPI technology.

5.5 The action research intervention

In this section, I summarize the contribution of paper 4 (Nielsen and Tjørnehøj, 2009).

Purpose:
The purpose of this study was dual, as for all action research. We wanted to contribute to the solution of the problems in *SmallSoft* on how to organize an effective SPI effort. We also wanted to understand knowledge sharing in SPI better and to find out how social networks analysis could be utilized for this.

Approach:
We performed an iterative social network analysis of the communication and knowledge-sharing networks of *SmallSoft*, mapping the results graphically in networks models. We found a misfit between the networks and the formal centralized improvement strategy that *SmallSoft* had followed previously and we could describe the misfit and findings in detail based on the mappings. The analysis was presented to the management of *SmallSoft* and led to a detailed discussion of views on the situation and of the future organization of SPI in the firm. Based on this negotiation, they designed and decided a new SPI organization fitting the networks. Further description of the approach of the study is set out in section 4.2.5 as the embedded action research intervention.

Findings and results:
We found that the analysis and the resulting network models were useful both to understand the knowledge-sharing networks and when communicating the results to the managers. These also supported the managers in negotiating the situation and

deciding on the future SPI organization. Also management found it useful as a kind of mirror in which they could see their own organization in a new light.

It was evident in the study that communication and knowledge sharing is an important integral part of SPI that follows patterns other than the official channels. It is important to understand these networks as they can hinder or promote SPI efforts. This can be done through social network analysis carried out as we did. This low budget approach is well suited for small firms since they are less likely to choose a formal centralized SPI strategy. The approach provides insights in the underlying social networks that are an important part of the infrastructure of informal SPI. It is likely to be useful to other small organizations.

Contributions:
The paper recognizes communication and knowledge-sharing networks as an important integral part of SPI and suggests that it is important to understand these to promote successful SPI efforts. However, the main contribution is providing an example of how to use social networks analysis in SPI, and proving it useful when investigating knowledge sharing and communication in SPI in smaller firms.

6 Discussion of the results and implications

The main finding from my analysis of the *SmallSoft* case was that the SPI technology adoption drifted in unpredictable directions. In practice, SPI technology is drifting as described in the case in paper 2 (Tjornehoj and Mathiassen, 2008a, section 5.3). I have investigated this further in my research (papers 2, 3, and 4). In this section the results will be discussed, organized in five themes of SPI practice, and the implications for SPI research and practice will be addressed.

6.1 SPI practice characteristic

I have chosen the five themes because they characterized the drifting SPI practice of *SmallSoft*. Two of the themes are already main topics of papers 2 and 3 (Tjornehoj and Mathiassen, 2008a, section 5.3). The other themes have grown in importance to me during my work on the thesis and are thus added here.

The first three themes address conditions for drifting SPI practice while the others address aspects of how to act in drifting SPI practice. Together they outline a profound understanding of important characteristics of SPI practice based on empirical findings. This understanding may lead to better advice for SPI practitioners than is currently provided in the dominant SPI theory.

The themes are:

- dependent on the production network
- sensitive to dynamic environments
- longitudinal
- shaped between control and drift
- improvisational.

For each theme I first present my findings and discuss SPI theory in accordance with the theme, and secondly I state my contribution.

6.1.1 Dependent on the production network

This theme addresses the role of SPI practice in a software organization, especially how it is related to the main business of the organization: software production.

In paper 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) we account for the existence of two networks[5] in *SmallSoft:*

> …the relatively stable and powerful *production*-network in which managers and software developers across *SmallSoft*'s three departments developed new solutions in response to customer requests. (Tjornehoj and Mathiassen, 2008a, p. 75)

and

> …the less stable and weaker *improvement*-network through which a small group of different actors over time attempted to improve practices in the production-network through the adoption of new development technologies. (Tjornehoj and Mathiassen, 2008a, p. 75).

Through the process analysis of the paper we describe how the production network and the improvement network interact in and how the production network dominates the 10 years of adoption of SPI technology in *SmallSoft.* We show how the successful improvements often are the ones driven mainly by the production network and its pertinent needs. We also show that if the production network is successful and provides surplus then investments in the improvement network is more likely. That is, until the demand overheats the production network and requires all resources. In *SmallSoft* the improvement network depends highly on the state of the production network (Tjornehoj and Mathiassen, 2008a, p. 83). Having realized this, in the final SPI initiative that we report on, the SPI manager in *SmallSoft* aligned the interests of the improvement network with the production-network by forming self-governing cross-firm process improvement teams (PITs) involving all employees.

In contrast to this, SPI theory advises that SPI should be organized separately from the production. A group (the SEPG) (Humphrey, 1989, p. 287) of dedicated change agents is to initiate, design and drive the improvements. The group forms an independent change organization that will only be informed about system development from outside by the engineers engaged in production. The improvements have to start at the top and be supported by committed top management to create the momentum needed (Humphrey, 1989, p. 19). Also the effect of the improvement effort is measured by an external norm, independent of the software production in the organization. Most SPI literature takes this for granted. This applies both to surveys reporting adoptions of SPI technology (Haley, 1996, Hollenbach et al., 1997, Hideto et al., 2006) and to studies of success factors in SPI (Herbsleb et al., 1997, El-Emam et al., 2001, Wilkie et al., 2005). However,

[5] In the sense of actor networks (Callon, 1986, Latour, 1987, Walsham, 1997).

this separate organization of the improvement staff and activity will hinder the improvement network from aligning with the production network because of lack of shared activities, knowledge and interests.

My finding that the improvement network is inherently dependent on and can benefit from being aligned and integrated with the production network questions the benefits of separate organization. A separate organization will make the alignment unlikely to happen. The detachment of the production and the improvement efforts constrains potential synergy from aligned interests, from employees feeling ownership for the improvements and from opportunities offered from the production network.

Aaen (2003) argues theoretically that the original SPI approach (Blueprint SPI) externalizes process knowledge and separates process design from use. The approach thereby risks seeing the process models as ends rather than means to improvement, risks gold-plating of the processes and risks plain useless changes. Our empirical findings support Aaen's theoretical finding (2003). We described how the production network continuously helped focus the improvement network on pertinent needs and provided powerful feedback to planned or implemented improvements.

I suggest that aligning[6] the improvement network with the production network will allow SPI to be fueled by the most powerful network of the organization. This could ease the problems of lack of resources and failed investments and help ensure that planned improvements fit the firm's reality. Aligning the two networks would, among other things, involve acting in the interests of the production network (Iversen et al., 1998b) and furthering cross-network activities and knowledge sharing (Nielsen and Tjørnehøj, 2009). Extensive user participation in the design and implementation of improvements (Aaen, 2002) or integration of improvement initiatives in system development practices (Börjesson and Mathiassen, 2004) are possible roads to this.

6.1.2 Sensitive to dynamic environments

This theme addresses how the increase in environmental dynamics of organizations has changed the premises of SPI practice and thus must change the practice itself.

The sensitivity to dynamic environments is an important driver in technology adoption. In the *SmallSoft* case, both challenges and opportunities offered by the dynamic environment were acting during the adoption, as described in papers 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) and 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4). We saw how market fluctuations reduced the ability to invest in SPI both when *SmallSoft* had to downsize and when the firm

[6] In the sense of aligning interests and networks in actor network theory, see (Callon, 1986, Latour, 1987, Walsham, 1997).

experienced unexpected increased sales, and how the opportunity of action learning changed the SPI strategy. Paper 4 (Nielsen and Tjørnehøj, 2009, section 5.5) reports in detail on one such example of an externally provided opportunity. In paper 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4), we especially describe how *SmallSoft* reacted to and utilized the dynamics of the environment through a flexible, improvisational behavior that allowed for adapted and useful solutions.

Others have also found that small firms are sensitive to highly dynamic environments (Mathiassen and Vainio, 2007) and Conradi and Fuggetta (2002) state that business and market turbulence can be a hindrance when adopting SPI technology. However, the dominant SPI theories do not have an answer to this challenge. To the contrary, a static strategy for SPI is promoted by the norm-driven approaches (Arent, 2000), which include, among others, BOOTSTRAP (Kuvaja, 1999), the ISO9000 series (Hoyle, 2005) and CMM(I) (Paulk et al., 1993, The-CMMI-product-team, 2001, 2002). The goal of the improvement activity is compliance with rather static norms. Success is when assessments show an increasing maturity according to the norm. This kind of SPI strategy does not allow for awareness of and adaptation to a dynamic firm environment unless this environment coincidentally is mirrored in the norm.

A commercial SPI business has formed around the norms. The underlying perception that the best practice of system development is rather general across industry and time keeps the norms static. Organizations certified according to a norm of course support this, since every update of the norm can be costly for them. Thus there is a major risk of growing misfits between the increasingly dynamic and unpredictable environments and the rather static norms. Adopting an inappropriate norm could lead to failed investments and unfeasible improvements even though these may be successful according to that norm. Likewise, opportunities offered by the dynamic environment will rarely fit the norm and they will probably be wasted.

Since traditional SPI demands many, expensive and long-term improvements (Aaen et al., 2001) for most firms, the time period without acting on dynamic environments is likely to be long. The separate organization of SPI through the SPEG as described above (Humphrey, 1989, p. 287, section 6.1.1) just adds to the static nature of the SPI strategies. A static group of full-time change agents are shielded from the environments of the organizations by top management and by not practicing system development themselves.

The static aspect of the dominant SPI approaches has been criticized for not reflecting environmental change (Ward et al., 2001), for focusing on process stabilization and refinement when fast-paced environmental change demands product innovation (Conradi and Fuggetta, 2002), for assuming a relatively high level of stability in the environment (Börjesson and Mathiassen, 2005), and for emphasizing process control more than building abilities to respond to environmental change (Aaen et al., 2007). Aaen (2003) states that SPI theory

promotes systematic planning for the expected rather than the unexpected, and argues that this would rarely be the best answer, given the complexities and uncertainties of software projects.

The SPI adoption of *SmallSoft* was certainly complex, unpredictable and characterized by dynamic environments. According to Ciborra, dynamic environments can lead to drifting technology. He suggests taking advantage of this by supporting human innovation and by facilitating the cultivation and hosting of technology instead of trying to plan or design it (Ciborra, 2002). In the *SmallSoft* case, major and important changes in SPI strategies were brought about by this kind of behavior. Some examples are mentioned above. In particular, we found that deliberately cultivating the ability to improvise in appropriate ways was indeed helpful when adopting SPI technology in dynamic environments.

The lack of ability of small organizations to withstand dynamic environments is an obstacle when adopting traditional SPI technologies since these imply long-term plans and fixed goals defined through a norm (Aaen et al., 2001). Since dynamic environments are common, in particular for small organizations (Holmberg and Mathiassen, 2001), I suggest embracing this dynamic as an advantage for SPI technology adoption. This will allow the adopted SPI technology to be fitted to the actual situation and the adoption to benefit from possibilities offered from outside. It will be likely to foster flexible and integrated improvements that are beneficial for the organization also in the short term.

Some resemblance to the agile trend in software development (Beck et al., 2001) is obvious, as this too promotes embracing change. Embracing the dynamic environments in SPI will involve short-term changes and evaluations of usefulness, allowing changes in the environment to be accounted for and utilized continuously. This will call for more flexible approaches to SPI as suggested by (Börjesson and Mathiassen, 2005) and (Aaen et al., 2007).

6.1.3 Longitudinal

This theme addresses the longitudinal nature of SPI practice as a key to understanding the web of learning that leads to improvement.

The basic analysis on which both papers 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) and 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4) build is organized as a longitudinal interpretive study. Among other things this means focusing on temporal interconnectedness: “Antecedent conditions shape the present and the emerging future…Thus history is not just an event in the past but is alive in the present and may shape the future.” (Pettigrew, 1990, p. 270). History is here understood as more than events in chronological order. In both studies it became increasingly apparent how history thus shaped the present, even though from the start we did not anticipate direct connections between the different SPI efforts. One of the employees interviewed stated that, even if the improvement efforts at first

sight might look like failures, the firm had changed and learned a lot. He also pointed out that the improvements in the organization were noticeable, even though the efforts evaluated one by one in a short-term perspective were either failures or successes that soon evaporated. We realized that a web of events, individual and collective learning, new personal practices and tools, communications and discussions and management actions, bit by bit and building on each other over time, had moved the organization forward and created new practices.

One of the key principles of the dominant SPI theory is that improvement is continuous (Humphrey, 1989, p. 19). It is not a one-shot effort, but takes continuous learning and growth. Since people and problems are in a constant flux, Humphrey suggests periodic adjustments of task and relations (Humphrey, 1989, p. 20), but advises doing so in a disciplined way in stable periods to allow for focusing on the processes and not on the immediate problems. The goal is still an orderly coherent improvement framework (Humphrey, 1989, p. 21). Bits and pieces – or an incoherent patchwork (p. 21) – does not count. Continuous improvement is measured according to the norm and only improvements that are part of an orderly improvement framework count.

Mathiassen et al. (2002) support the dominant SPI theories' assumption that SPI is inherently continuous as "there are always new problems and challenges, and solutions to old problems must be maintained and further developed" (Mathiassen et al., 2002, p. 17). They find that continuous improvement has to be stepwise, supported by top management commitment to keep momentum, and conducted by a sustainable improvement organization.

In the *SmallSoft* case, we found many small changes that either cleared the way for improvements or gathered and gained power over time to eventually improve practice. Many of these would be seen as insignificant and even unwanted by the dominant SPI theories. When an organization does not value these micro changes, some potential for grown improvements is lost and they risk stunting the development of an improvement culture. The idea of history actively shaping the present and future means that even ignored changes that are insignificant according to the norm will continue to impact future improvements either positively or negatively. Assessing organizations according to a norm leads to a risk of ditching improvements that might have some potential because they are regarded as failures. In *Smallsoft* we found that failed improvement efforts laid the foundations for the improvements, among other things through learning, new shared understanding and reusable artifacts.

Both Mathiassen et al. (2002) and Humphrey (1989) use the concept "continuous SPI", addressing that an organization should keep taking stepwise SPI action, whether because "it takes time to climb the ladder of maturity" or because "new challenges and problems arise". Based on my research, I suggest longitudinal SPI as a richer and more appropriate way of thinking than continuous SPI. It is a way to

emphasize the web of learning, actions and artifacts that, through history, lay the foundations of any improvement effort that an organization plans. Acknowledging this may change the perception of what is, or can lead to, an improvement.

To go beyond the simplistic and common understanding of change as rationally planned and implemented and to grasp the real complexity of practice, dominant SPI theories need to be supplemented by other lines of theory dealing with, for example, the social aspects of organizations (Nielsen and Nørbjerg, 2001, Ciborra, 2002), knowledge (Mathiassen and Pourkomeylian, 2003) and learning (Fichman and Kemere, 1997).

6.1.4 Shaped between control and drift

This theme outlines a philosophy for SPI practice that fits the conditions for SPI practice that I have found through my work.

Paper 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) argues that the adoption of SPI technology in *SmallSoft* was shaped between control and drift. We found both elements of control and drift that had beneficial impacts on the adoption. The elements were interacting, with their relative dominance shifting.

The SPI theories and models that were introduced offered control approaches, facilitated knowledge sharing and learning and helped management to set the direction. This way the control elements framed the collaborative experimenting and learning, and kept the improvement network alive. The continued control efforts kept *SmallSoft* vigilant paper 2 (Tjornehoj and Mathiassen, 2008a, p. 85) by insisting on and pushing the organization toward change.

Drifting at the same time helped to ensure the adaptation of SPI technology to the firm's realities, to exploit human creativity and innovativeness, and to handle lack of resources and knowledge. Unexpected opportunities from outside the firm and everyday coping, bricolage and improvisation by employees, were important during the adoption (see paper 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4)).

Together the two shaped the adoption process, interacting with, balancing and moderating each other. When the control elements balance the drift elements they ensure the effectiveness and efficiency of the practice, for example, by avoiding over-improvising in routine situations and by providing means for coordination (not only traditional planning, but less rigorously in goal setting and knowledge sharing). When the drift elements balance the control elements they ensure adaptation of the models, plans and technologies to the real life of the organization. For example, unfeasible planned improvements are ignored or changed, new improvements are sparked by improvisation as a reaction to the dynamic environment, and the adoption process itself is molded to fit the production network's situation. Together this secured a unique solution for *SmallSoft.*

From the dominant SPI theory we already know a lot about control elements and how to utilize control. For example, the core idea of CMM (Humphrey, 1989) and IDEAL (McFeeley, 1996) is for management to control the continuous improvement activity of the firm. In general, SPI literature focuses on prescribing how best to control and measure the working processes of a firm in accordance with a norm. Drift is at best addressed in explaining failed efforts or in some cases as supplementary to the main drivers of improvement.

According to Ciborra (2002), this control view is widespread in the field of IS. It is based on a rational worldview, in which managers understand and plan events by applying simplistic theoretical models to decisions and practices. Ciborra finds that this detachment from the real world causes a crisis in the field and he suggests firms should discard control and organize for drift to stay innovative and competitive. CMM is a clear example of what Ciborra wants to avoid (Ciborra, 2002, p. 19).

Drift describes how side effects, bricolage, hacking, formative context, and people's everyday coping in an increasingly complex and unpredictable world make reality drift away from plans, thereby opening the way for options and innovations that otherwise would be unthinkable.

Ciborra finds control and drift to be paradigms and thus irreconcilable. However, we found that not only did elements of both contribute positively to the adoption of SPI technology in *SmallSoft*, they also acted together, balancing and moderating each other beneficially (Tjornehoj and Mathiassen, 2008a, section 5.3). My research suggests that negotiating the adoption process of SPI technology between the two will help in utilizing the full potential of improvement of software practice of an organization.

By utilizing control elements such as internal assessments according to a well-known norm, management can set the direction, vitalize and push the adoption of SPI technology. By not acting, they risk that the production network will petrify in an inappropriate practice. However, if management insists on a pure control approach without being open to the backtalk from the situation (often perceived as drift), they risk missing the full learning and innovation potential offered by the situation. On the contrary, they should cultivate the organization's ability to take advantage of drifting to moderate the adopted SPI technologies.

6.1.5 Improvisational

This theme goes into further detail about how to deal with the conditions under which SPI is practiced. One way of elevating the potential of drift is by cultivating the organization's capacity for improvisational action.

In immediate continuation of *SmallSoft's* sensitivity to dynamic environments comes the finding that the SPI practice was to a large extent improvisational, as

described in paper 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4). Improvisation can happen when something unexpected occurs, for which the organization has no plans or procedural memory (Gersick and Hackman, 1990). If this occurrence is perceived to demand such speedy action that planning and action have to converge, this action is called improvisation (Cunha et al., 1999).

We found improvisations at all levels of *SmallSoft*; some helpful for the adoption of SPI technology, others not. Improvisations helped employees perform even though resources were scarce and unanticipated challenges arose. Improvisation also resulted in improvements fitted to practice and the firm took advantage of opportunities offered from outside through improvisation. However, the improvisational culture of *SmallSoft* in some cases led to over-improvising in situations when time and resources actually could allow for planning, knowledge search and orderly action. We also saw instances of improvisation that was not in line with the interests of the firm since appropriate leadership and coordination was lacking. In summary, we found that the improvisational culture of *SmallSoft* was a great strength in the dynamic environment, provided that the improvisations addressed appropriate challenges and were supported and coordinated to ensure benefits for the firm.

In the dominant SPI theories, planning based on assessment is emphasized as an immensely important principle for software process change. "If process improvement is not rigorously planned and tracked, it will not happen" (Humphrey, 1989, p. 23). It is also stated that the key elements of change are planning, implementation and communication, and that it is important to "maintain a continuous stream of actions and successes" (Humphrey, 1989, p. 32). To reassure the employees and to keep their support, it is "essential to have public plans, periodic progress reports and early demonstrations of success" (Humphrey, 1989, p. 32). Here, success means according to the plans and the norm. The IDEAL model (McFeeley, 1996) prescribes well planned and rigorously conducted learning cycles that implement the continuous stepwise improvement prescribed by the CMM (Paulk et al., 1993). The CMM describes how the organization stepwise instals a substantial procedural memory until all software processes are defined and measured. Mathiassen et al. (2002) supports this view of continuous improvement as being well planned, stepwise, and supported by management funding and a sustainable improvement organization.

Following this advice will just not allow for improvisational actions since planning should be carried out before action and since the openness towards surprises is systematically reduced. The idea of a defined and measured process is to reduce uncontrolled actions by having a substantial procedural memory on which to draw. This way fewer occurrences will be perceived as unexpected and risk being more or less ignored.

As described in section 6.1.2, the dominant SPI approaches and organization are not easily adjustable to fit surprises. This applies to the concrete plans as well, as they are based on rather extensive assessments and static norms. Whether the unexpected comes from within the organization or from dynamic environments (See section 6.1.2) makes no difference. The dominant SPI approaches diminish the conditions for improvisation; experimental culture, a minimal structure and a low procedural memory (Cunha et al., 1999). Ciborra (2002) states that it is through bricolage, hacking and improvisation by individuals that organizations adopt technologies and achieve important innovations. Complying with the dominant SPI theories thus seems to increase the risk of potential for innovations remaining untapped.

My research suggests taking advantage of the improvisational power of an organization. To do so we need to address improvisation as a competence that we should cultivate. We need to grow an experimental culture, but also to implement leadership and minimal structures to support and coordinate the improvisational actions of the firm. With regard to procedural memory, it is important how we perceive the procedures. They can be taken as an outset for improvisation (Cunha et al., 1999, Aaen, 2003) and thus be beneficial for an improvisational organization.

6.2 Summary of results

I have answered my research questions through my research contributions and the discussions in this summary.

I have addressed RQ1 "what is the problem with the dominant SPI theories understanding of SPI practice?" in my literature study paper 1 (Hansen et al., 2004a, sections 2.1 and 5.2) and again in the discussion of all five themes. Some of the problems with the dominant SPI theories understanding of SPI practice do stem from the shape of the research field (Hansen et al., 2004a). The single-loop learning cycle of the field, the dominant status of the CMM, and the SPI industry that has formed around the models, conserve the underlying assumptions of the original model in the field and hinder alternatives being developed and tested (Hansen et al., 2004a, section 6.1). In my discussion I argue for each theme that the dominant theories do not have appropriate answers to the challenges of and the conditions for SPI practice. The dominant SPI theories instead promote:

- separate organization of the SPI network which hinders alignment with the production network
- static SPI strategies which do not allow for awareness of and adaptation to a dynamic firm environment, whether it brings new challenges or possibilities
- a limited view on continuous change which risks missing the potential of grown improvements, ditching helpful improvements and stunting the development of an improvement culture

- control approaches which miss the potential of drift both on its own and as a moderator of control.
- models that diminish the conditions for improvisation which miss the potential for innovation.

While RQ2 "what characterizes SPI practice?" was the driving interest in the case study (papers 2 and 3, see section 4.2.4, 0 and 5.4) and the action research intervention (paper 4, see sections 4.2.5 and 0), RQ3 "what new theories explain this practice better?" was central to both interpretations of the case study.

Answering RQ2, I found in the case study that overall SPI practice is characterized by drifting SPI technology. Plans are made, control is exercised, but SPI technology drifts in unpredictable directions anyhow (paper 2, section 5.3). In the discussion I point to three important conditions that characterize this drifting SPI practice (See sections 6.1.1–6.1.3). Drifting SPI practice is:

- inherently dependent on the production network
- sensitive to dynamic environments
- in nature longitudinal.

These conditions impose increased complexity and dynamics into SPI practice. To embrace these conditions, in order to benefit from them, adoption of SPI technology can be negotiated between control and drift (paper 2, section 5.3.). One important aspect of this negotiation is to cultivate the organization for improvisational action (paper 3, section 5.4.). I argue that drifting SPI practice has to be:

- negotiated between control and drift
- improvisational.

Answering RQ3 lay directly in these last two points. Drift theory is better suited to explain SPI practice than the current dominant SPI theories. However as discussed in section 6.1.4, control theory plays its own important role in SPI practice.

6.3 Implications for the SPI field

6.3.1 Implications for SPI theory

Dominant SPI theories build on models that are far too simple to capture the wealth of actors, interests and conditions that act in an adoption of SPI technology. They are also too rigid to allow for the flexibility needed to meet the challenge from dynamic environments and changes in the production network. And even though the models address continuous change, they do so with a narrow focus on orderly change towards a coherent improvement framework, unaware of how the full complex history shapes the present and the future. In summary, the dominant SPI

theory is constrained by ignoring important aspects of the adoption process and by being inflexible and narrowly focused.

Together the five characteristics above suggest that a richer understanding of SPI practice in the SPI research field is needed to meet the challenge of dynamics and complexity. My research has provided some new understanding of SPI practice, but much research still needs to be done on validating and extending this new knowledge. I will first address appropriate research topics for this then appropriate approaches.

When acting on the challenge of dynamics and complexity, Ciborra (2002) advocates leaving the control view behind. However, we found that the control approaches impacted positively on the SPI adoption when interacting with and moderating the drifting, and, by the way, vice versa. Based on my discussion (section 6.1) I suggest exploring radical new ways to deal with improving practice and capabilities of firms in line with drift theory. I use the word capabilities here to underpin that even the focus of what is important to improve should be questioned. This would include more practice studies. However, "shaped between control and drift" suggests including the knowledge from the per se dominant SPI theory, but reinterpreting its recommendations in the light of the much richer understanding of the adoption of SPI technology. The goal would be situated, flexible and adaptive approaches that exploit the possibilities of drifting (see the five themes above in section 6.1).

According to the IS research framework (Braa and Vidgen, 1999), understanding is best reached through case-study approaches (section 4.1.1). Thus more case studies are recommended. However, we also need to take this knowledge of SPI practice further into more prescriptive theory in order to advise practitioners better. Experimenting with alternative SPI approaches that fit the new knowledge of SPI practice and intervention into this practice will furnish this.

To develop and integrate the new understanding of SPI practice, contributions from all types of research in the IS research framework (Braa and Vidgen, 1999) will be beneficial. As they utilize very different dynamics (Braa and Vidgen, 1999, p. 27) in order to reach very different outcomes, the resulting knowledge is likely to supplement and enhance each other.

In summary, a more profound understanding of SPI practice adding to my research is needed to inform the research of the field. The research needs to be refocused to fit this understanding. This means both studies on drifting SPI and on reinterpreting control in that same context. Research of all kinds is needed to integrate the new understanding of SPI practice as drifting practice into the SPI research field (Braa and Vidgen, 1999). Collaborative practice research and other multi-disciplinary research approaches (Mingers, 2001, Mathiassen, 2002) could be useful since they encourage theory building on knowledge of practice.

The dominant SPI theories and the whole industry that has formed around these norms, including the research closely connected to the industry, has been challenged before, theoretically by Aaen (2003), Ngwenyama and Nielsen (2003) and Rose et al. (2008), and now also empirically through my work. But obviously firms in the industry may have no interest in a changing their business. I thus find it difficult to imagine the SPI research field changed as described. Instead I speculate that it may split into more research directions or even new fields.

6.3.2 Implications for SPI practice

In the light of drifting SPI as discussed in the five themes above, organizations that want to adopt SPI technology will face dynamic environments and complexities beyond what is described in SPI theory (section 6.1). This need not be new to SPI managers since they are part of practice, but the news is that coping with the challenge cannot be dealt with through the dominant SPI theories (section 6.1) alone.

Organizations will need to negotiate control and drift when adopting SPI technology in order to improve their practice. This means that they need to acknowledge the complexity of the challenges and to cultivate the organization's ability to display both control and drift capabilities when improving. Advice can be found in paper 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) on how to negotiate technology adoption constantly *between control and drift* and in paper 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4) on how to utilize improvisational action in the adoption. The advice of this last paper is the most practical and discusses the following aspects that managers should consider:

- Cultivate improvisations
- Facilitate deliberate improvisations
- Provide support structures for improvisation
- Exercise leadership when improvising.

Aligning the production network and the SPI network will help to create momentum in the improvement work and will ensure feasible improvements as discussed in section 6.1.1. Problem-driven SPI (Iversen et al., 1998b) and integration of process design and use (Aaen, 2003) are two possible ways to do so. Embracing dynamics from the environment will bring possibilities and secure adaptation of the adopted SPI technology to the firm's reality. Some of the new research on agile SPI might provide good advice on that (Börjesson and Mathiassen, 2005, Aaen et al., 2007). The longitudinal perspective on improvements will help the organization to understand the situation better and to draw from all sources when improving. This is discussed in section 6.1.3. Both actively cultivating the organization for improvisational action (Tjornehoj and Mathiassen, submitted 2007, section 5.4) and utilizing social network analysis in order to inform decision making in the adoption process (Nielsen and Tjørnehøj,

2009, section 5.5) could also be helpful when exploiting the drifting reality of SPI practice.

In short, organizations need to re-interpret the SPI theory into their special situation, aligning the production network, embracing dynamic environments and keeping a "true" longitudinal perspective on improvements. In addition, they need to cultivate their organization for drift capabilities, notably improvisation.

7 Conclusion and future research

This chapter concludes the PhD thesis with a summary, a discussion of limitations and suggestions for future research.

7.1 Summary

In this PhD thesis I have presented my research on SPI practice. I have argued that there is a gap between the understanding of SPI practice in the dominant SPI theories and SPI practice. My research questions (section 2.5) explore this gap by investigating

- problems with the existing understanding
- the SPI practice
- theories that explain practice better.

I have studied the literature of the SPI field in order to determine the shape of the field (paper 1, Hansen et al., 2004a, see section 5.2). This study underpinned a need for more independent and reflective research on SPI. I then studied a small software firm both through action research on knowledge networks (paper 4, Nielsen and Tjørnehøj, 2009, see section 5.5) and through a longitudinal interpretive case study (section 4.2.4). The first analysis focused on how adoption of SPI technology was shaped between control and drift (paper 2, Tjornehoj and Mathiassen, 2008a, see section 5.3). The second focused on the role of improvisation in this process (paper 3, Tjornehoj and Mathiassen, submitted 2007, see section 5.4).

In order to answer my research questions I have discussed five themes that characterize SPI practice:

- dependent on the production network
- sensitive to dynamic environments
- in nature longitudinal
- shaped between control and drift

- improvisational.

I found that the dominant SPI theories do not have appropriate answers to the challenges of SPI practice. SPI practice is characterized by drifting SPI technologies and it will be beneficial to negotiate control and drift when adopting SPI technologies to take advantage of the drifting forces. Improvisation is one way to do so. Thus these theories – drift and improvisation –explain practice better than the dominant SPI theories.

The implication for SPI research of the new understanding of SPI practice is that we need to reinterpret the dominant SPI theory in the light of a much more profound understanding of the complexities of SPI practice, but also that we need to explore radical new ways to deal with improving practice. The implication for practitioners is that they will have to leave the vicious cycle of control (Ciborra, 2002) in which they may be caught and start negotiating control and drift instead when adopting SPI.

This new knowledge is relevant to the software industry, helping it to understand better the challenges of SPI practice and to take appropriate action when improving system development practice. I have published two of the papers in a form that is better suited for practitioners , in the book, *Beyond Conventional Software Process Improvement* (Nielsen and Kautz, 2008). Thus my first research goal, "contribute relevant knowledge to the software industry in order to support their efforts to improve their system development practice", is achieved.

The knowledge is based on independent and reflective research (section 4.2). The researchers were outsiders in the case study and by no means dependent on the studied firm or other organizations that could have influenced the results. The PhD study is reflective by using theory for analysis and generating new theoretical understandings and by "challenging basic taken-for-granted assumptions" (Hansen et al., 2004a). Thus my second research goal, "contribute to the research field of software process improvement with independent and reflective research" is achieved.

7.2 Limitations

According to Braa and Vidgen's IS research framework (1999), as well as collaborative practice research (Mathiassen, 2002), it is desirable to combine research approaches to balance the relevance and rigor of the research and thus raise the quality of the results. This PhD study is based solely on a longitudinal interpretive case study, that is interpretive research in the IS research framework (lower left point). This is an obvious limitation of my work. However, this is a rather work-intensive, skill- and time-demanding research approach and a PhD project is limited in time, which put natural limits on supplementing the approach.

The study in itself is limited by being a single case study. Pettigrew (1990) suggests adding studies of other similar cases to build a multi-case study, so that generalizations across cases could be drawn. Not having done this limits my results to what is appropriate for a single case study.

The fact that much data was retrospective is also a limitation. Interviewee reflecting retrospectively on events might forget or rationalize what happened. Retrospective data are commonly utilized within IS research and to minimize the problems we have been looking for data from complementary data sources to support each other.

Seen in isolation, the research in my project has the abovementioned limitations. The limitations are the reverse side of the coin of a unique study possibility combined with limited project time. Even though the research that is reported in this research project is somewhat narrow, it cannot be seen completely on its own. First, it was framed by the SPK project that was organized as collaborative practice research. Data from some of the SPK research has served as data in my study, and the other research approaches, topics and results have served as part of my prior knowledge when analyzing and interpreting the case.

Second, my research was initiated to supplement the SPI research already done. A large amount of research from the left side of the IS research framework (Braa and Vidgen, 1999, p. 31) has been carried out, resulting in numerous prescriptive and some descriptive contributions. My research can be seen as supplementary to their results, doing research on the same topic but from a different perspective and with a different research approach. Thus awareness of the body of knowledge in SPI, both the original and the Scandinavian tradition (including the SPK project), helps to balance this research.

7.3 Future research

Above I have pointed to types of research and research topics that would be appropriate in the field of SPI in the future in order to integrate this new understanding of SPI practice in general (section 6.3.1). Here I will sketch my own future research interest in the SPI field. Beginning with the research reported in paper 3 (Tjornehoj and Mathiassen, submitted 2007) I find it obvious to investigate further the role of improvisation in SPI practice.

I think we still need to learn more about improvising in SPI practice in order to advise practitioners on how to benefit from improvisational action. I would like to supplement my understanding from the case study with more involvement in practice, either through field experiments or action research. We often learn much about practice by aiming to change it. Since I do not think that my new understanding provides the basis for designing field experiments I prefer action research. It takes its outset in problems experienced by practitioners in an organization, thus the exact content of the research cannot be planned. However,

appropriate selection of industry partners will help putting improvisational action in focus. A partner organization experiencing dynamic environments but lacking resources to handle this challenge could be a good choice (since deliberate improvisation may be an appropriate answer to its problems).

Examples of new research questions that puzzle me are:

- Can improvisation be chosen and utilized deliberately in SPI practice in an organization? And if so, how is this done?
- How does management support improvisational action in SPI practice? And how do they cultivate an organization for improvisation?
- How is improvisation carried out by SPI practitioners? What does it take to be able to improvise?
- What are the pitfalls of improvisation in SPI practice and how can they be avoided?
- What are the outcomes of improvisation in SPI practice?

The questions address improvisation from when it is initiated or decided, through practicing it, to the actual outcome. They cover more levels of analysis by focusing both on individuals and on the organization and they address how improvisation could or should to some extent be controlled to become beneficial for an organization.

The questions express my broad interest in the topic of improvisation in SPI practice.

References

Aaen, I. (2002). Challenging software process improvement by design, in *the 10th European Conference on Information Systems (Gdansk. Poland, 2002)*, pp. 379-390.

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

Aaen, I., Arendt, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement, *Scandinavian Journal of Information Systems* 13: 81-101.

Aaen, I., Börjesson, A. and Mathiassen, L. (2007). SPI Agility: How to Navigate Improvement Projects, *Software Process: Improvement and Practice* 12(3): 267-281.

Allison, I. and Merali, Y. (2007). Software process improvement as emergent change: A structurational analysis, *Information and Software Technology* 49(6): 668.

Arent, J. (2000). *Normative Software Process Improvement*, Department of Computer Science. Aalborg University.

Batista, J. and Figueiredo, A. D. D. (2000). SPI in a Very Small Team: a Case with CMM, *Software Process: Improvement and Practice* 5(4): 243-250.

Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. (2001). Manifesto for Agile Software Development, http://agilemanifesto.org/.

Börjesson, A. and Mathiassen, L. (2004). Successful Process Implementation, *Software, IEEE* 21(4): 36.

Börjesson, A. and Mathiassen, L. (2005). Improving Software Organizations: The Agility Challenge, *Information, Technology & People* 18(4): 359-382.

Braa, K. and Vidgen, R. (1999). Interpretation, intervention, and reduction in the organizational laboratory: a framework for in-context information system research, *Accounting, Management and Information Technologies* 9(1): 25.

Brodman, J. G. and Johnson, D. L. (1994). What Small Businesses and Small Organizations say about the CMM, in *16th International Conference on Software Engineering (ICSE-16) (Sorrento, Italy, 1994)*, IEEE Computer Society Press, pp. 331-340.

Brouse, P. S. and Buys, R. T. (1999). Affordable ways to improve application development, *IT Professional* 1(4): 47-52.

Callon, M. (1986). Some elements of a sociology of translation: domestication of scallops and the fishermen of St Brieuc Bay, in Law, J., (Ed.) *Power, action and belief: a new sociology of knowledge?* London: Routledge & Kegan Paul, pp. 196-223.

Casey, V. and Richardson, I. (2004). A practical application of the IDEAL model, *Software Process: Improvement and Practice* 9(3): 123-132.

Checkland, P. (1991). From framework through experience to learning: the essential nature of action research, in Nissen, H.-E., Klein, H. K. and Hirschheim, R. A., (Eds.) *Information Systems Research: Contemporary Approaches and Emergent Traditions,* North-Holland, Amsterdam, pp. 397-403.

Chelariu, C., Johnston, W. J. and Young, L. (2002). Learning to improvise, improvising to learn: A process of responding to complex environments, *Journal of Business Research* 55(2): 141-147.

Cho, S., Mathiassen, L. and Nilsson, A. (2008). Contextual dynamics during health information systems implementation: an event-based actor-network approach, *European Journal of Information Systems* 17: 614-630.

Ciborra, C. (2002). *The Labyrinths of Information: Challenging the Wisdom of Systems,* New York: Oxford University Press.

Conradi, R. and Fuggetta, A. (2002). Improving Software Process Improvement, *IEEE Software* 19(4): 92.

Cross, R. L. and Parker, A. (2004). *The Hidden Power of Social Networks: Understanding How Work Really Gets Done in Organizations,* Boston: Harvard Business Press.

Cunha, M. P., Cunha, J. V. and Kamoche, K. (1999). Organizational improvisation: what, when, how and why, *International Journal of Management Reviews* 1(3): 299-341.

Deming, W. E. (1982). *Out of the Crisis*, Cambridge, MA: Productivity Press.

Dion, R. (1992). Elements of a Process-Improvement Program, *IEEE Software* 9(4): 83-85.

El-Emam, K., Goldenson, D., McCurley, J. and Herbsleb, J. (2001). Modeling the Likelihood of Software Process Improvement: An Exploratory Study, *Empirical Software Engineering* 6(3): 207-229.

Fayad, M. E. and Laitinen, M. (1997). Process Assessment Considered Wasteful, *Communications of the ACM* 40(11): 125-128.

Fichman, R. G. and Kemere, C. F. (1997). Assimilation of Software Process Innovations: An Organizational Learning Perspective, *Management Science* 43(10): 1345-1363.

Gersick, C. J. G. and Hackman, J. R. (1990). Habitual Routines in Task Performing Groups, *Organizational Behavior and Human Decision Processes* 47(1): 65-97.

Golden-Biddle, K. and Locke, K. (1993). Appealing Work: An Investigation of How Ethnographic Texts Convince, *Organization Science* 4(4): 595-616.

Grechenig, T. and Zuser, W. (2004). Creating Organic Software Maturity Attitudes (COSMA) Selected Principles and Activities for Software Maturity in Small and Medium Software Enterprises, in *Fourth International Conference on Quality Software (QSIC'04) (Braunschweig, Germany, 2004)*, pp. 134-143.

Haley, T. J. (1996). Software Process Improvement at Raytheon, *IEEE Software* 13(6): 33-41.

Hansen, B., Rose, J. and Tjørnehøj, G. (2004a). Prescription, description, reflection: the shape of the software process improvement field, *International Journal of Information Management* 24(6): 457-472.

Hansen, B., Rose, J. and Tjørnehøj, G. (2004b). Prescription, description, reflection: the shape of the software process improvement field, in *UK Association of Information Systems Conference (Glasgow, Scotland, 2004b)*, Glasgow Caledonian University.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. (1997). Software Quality and the Capability Maturity Model, *Communications of the ACM* 40(6): 30-40.

Herbsleb, J. D. and Goldenson, D. R. (1996). A systematic survey of CMM experience and results, in *the 18th International Conference on Software*

Engineering (ICSE-18) (Berlin, Germany, 1996), IEEE Computer Society Press, pp. 323-330.

Hideto, O., Takashi, I. and Tetsuro, M. (2006). Practical Approach to Development of SPI Activities in a Large Organization: Toshiba's SPI History since 2000, in *the 28th International Conference on Software Engineering (Shanghai, China, 2006)*, ACM, New York, NY, USA.

Hollenbach, C., Young, R., Pflugrad, A. and Smith, D. (1997). Combining Quality and Software Improvement, *Association for Computing Machinery. Communications of the ACM* 40(6): 41.

Holmberg, L. and Mathiassen, L. (2001). Survival Patterns in Fast-Moving Software Organizations, *IEEE Software* 18(6): 51-55.

Horvat, R. V., Rozman, I. and Györkös, J. (2000). Managing the complexity of SPI in small companies, *Software Process: Improvement and Practice* 5(1): 45-54.

Hoyle, D. (2005). *ISO 9000 Quality Systems Handbook*: Butterworth-Heinemann.

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

Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991). Software Process Improvement at Hughes Aircraft, *IEEE Software* 8(4): 11-23.

Iversen, J., Johansen, J., Nielsen, P. A. and Heje, J. P. (1998a). Combining Quantitative and Qualitative Assessment Methods in Software Process Improvement, in *6th European Conference on Information Systems (ECIS) (Aix-en-Provence, France, 1998a)*, pp. 451-466.

Iversen, J., Nielsen, P. A. and Nørbjerg, J. (1998b). Problem Diagnosis in Software Process Improvement, in *International Federation of Information Processing (IFIP) (Helsinki, 1998b)*.

Iversen, J., Nielsen, P. A. and Nørbjerg, J. (1999). Situated Assessment of Problems in Software Development, *SIGMIS Database* 30(2): 66-81.

Johnson, D. and Brodman, J. (1996). Realities and Rewards of Software Process Improvements, *IEEE Software* 13(6): 99-101.

Kautz, K. (1998). Software Process Improvement in Very Small Enterprises: Does It Pay Off? *Software Process: Improvement and Practice* 4(4): 209-226.

Kautz, K. (1999). Making sense of measurement for small organizations, *IEEE Software* 16(2): 14-20.

Kautz, K., Hansen, H. W. and Thaysen, K. (2000). Applying and Adjusting a Software Process Improvement Model in Practice: The Use of the IDEAL Model in a Small Software Enterprise, in *22nd International Conference on Software Engineering (Limerick, Ireland, 2000)*, ACM Press, pp. 626-633.

Kautz, K. and Larsen, E. A. (2000). Diffusion theory and practice: Disseminating quality management and software process improvement innovations, *Information Technology & People* 13(1): 11-26.

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

Kelly, D. P. and Culleton, B. (1999). Process Improvement for Small Organizations, *Computer* 32(10): 41-47.

Kuvaja, P. (1999). BOOTSTRAP 3.0- A SPICE 1 Conformant Software Process Assessment Methodology, *Software Quality Journal*, (8): 7-19.

Latour, B. (1987). *Science in Action: How to follow Scientists and Engineers through Society*, Cambridge, MA: Harvard University Press.

Mathiassen, L. (1997). *Reflective Systems Development*, Department of Computer Science. Aalborg University.

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

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. (Eds.) (2002). *Improving Software Organizations: From Principles to Practice*, Reading, MA, Addison-Wesley.

Mathiassen, L. and Vainio, A. M. (2007). Dynamic Capabilities in Small Software Firms: a Sense-and-Respond Approach, *IEEE Transactions on Engineering Management* 54(3): 522-538.

McFeeley, B. (1996). *IDEAL: A User's Guide for Software Process Improvement*, Pittsburgh, PA: Software Engineering Institute

McGrath, J. (1982). Dilemmatics: The Study of Research Choices and Dilemmas, in Mcgrath, J., Martin, J. and Kulka, R., (Eds.) *Judgement Calls in Research,* Beverly Hills CA: Sage, pp. 69-102.

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

Mingers, J. (2001). Combining IS Research Methods: Towards a Pluralist Methodology, *Informations Systems Research* 12(3).

Newman, M. and Robey, D. (1992). A Social Process Model of User-Analyst Relationship, *MIS Quarterly* 16(2): 249-266.

Ngwenyama, O. and Nielsen, P. A. (2003). Competing Values in Software Process Improvement: An Assumption Analysis of CMM from an Organizational Culture Perspective, *IEEE Transactions on Engineering Management* 50(1): 100-112.

Nielsen, P. A. and Kautz, K. (2008). *Beyond Conventional Software Process Improvement,* Aalborg: Software Innovation Publisher.

Nielsen, P. A. and Nørbjerg, J. (2001). Software Process Maturity and Organizational Politics, in Fitzgerald, B. and Russo, N., (Eds.) *Realigning Research and Practice in Information Systems Development: The Social and Organizational Perspective, Proceedings of IFIP WG 8.2 Conference,* Boise, Idaho.

Nielsen, P. A. and Tjørnehøj, G. (2005). Mapping Social Networks in SPI, in *IFIP 8.6 Conference: Business Agility and IT Diffusion (Atlanta, GA, USA, 2005),* Springer Verlag, 2005.

Nielsen, P. A. and Tjørnehøj, G. (2009). Social Networks in Software Process Improvement, *Software Process: Improvement and Practice* online.

Paulk, M. C., Curtis, B., Chrissis, M. B. and Weber, C. (1993). *Capability Maturity Model for Software version 1.1*, Pittsburgh, PA: Software Engineering Institute

Pettigrew, A. M. (1990). Longitudinal Field Research on Change:Theory and Practice, *Organization Science* 1(3): 267-292.

Pino, F., García, F. and Piattini, M. (2008). Software process improvement in small and medium software enterprises: a systematic review, *Software Quality Journal* 16(2): 237.

Richardson, I. (2001). Software Process Matrix: A Small Company SPI Model, *Software Process: Improvement and Practice* 6(3): 157-165.

Richardson, I. (2002). SPI models: What characteristics are required for small software development companies? *Software Quality Journal* 10(2): 101-114.

Rose, J., aaen, I. and Peter Axel, N. (2008). Managerial and organizational Assumptions in the CMMs, in Nielsen, P. A. and Kautz, K., (Eds.) *Beyond Conventional Software Process Improvement*, Aalborg: Software Innovation Publisher.

Sakamoto, K., Kishida, K. and Nakakoji, K. (1996). Cultural Adaptation of the CMM: A Case Study of a Software Engineering Process Group in a Japanes Manufacturing Factory, in Fuggetta, A. and Wolf, A., (Eds.) *Software Process*: John Wiley and Sons, pp. 137-154.

Scott, L., Jeffery, R., Carvalho, L., D'Ambra, J. and Rutherford, P. (2001). Practical Software Process Improvement -The IMPACT Project, in *13th Australian Software Engineering Conference (ASWEC'01) (Canberra, 2001)*, IEEE, pp. 182.

Steel, A. P. C. (2004). Low-rigour, Rapid Software Process Assessments for Small Software Development Firms, in *2004 Australian Software Engineering Conference (ASWEC'04) (Melbourne, 2004)*, IEEE, pp. 323-334.

The-CMMI-product-team (2001). *Capability Maturity Model Intergration (CMMI-SM), Version 1.1, Continuous Representation*, Pittsburgh, PA: Carnegie Mellon, Software Engineering Institute

The-CMMI-product-team (2002). *CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1, Staged Representation (CMMI-SE/SW/IPPD/SS, V1.1, Staged), Technical Report of SEI*, Pittsburgh, PA: SEI

Tjornehoj, G. and Mathiassen, L. (2008a). Between control and drift: negotiating improvement in a small software firm, *Information Technology & People* 21(1): 69-90.

Tjornehoj, G. and Mathiassen, L. (2008b). The Role of Improvisation in Adoption of Process Technology in a Small Software Firm, in Nielsen, P. A. and Kautz, K., (Eds.) *Beyond Conventional Software Process Improvement*, Aalborg: Software Innovation Publisher, pp. 135-162.

Tjornehoj, G. and Mathiassen, L. (submitted 2007). Improvisation during Process-technology Adoption: A Longitudinal Study of a Software Firm, *Journal of Information Technology*.

Villalon, J., Agustin, G. C., Gilabert, T. S. F., Seco, A. D., Sanchez, L. G. and Cota, M. P. (2002). Experiences in the Application of Software Process Improvement in SMES, *Software Quality Journal* 10(3): 261-273.

Walsham, G. (1993). *Interpreting Information Systems in Organizations*, Chichester: Wiley.

Walsham, G. (1995). Interpretive case studies in IS research: nature and method, *European Journal of Information Systems* 4(2): 74-81.

Walsham, G. (1997). Actor-network theory and IS research: current status and future prospects, in Lee, A. S., Liebenau, J. and Degross, J. I., (Eds.) *Information Systems and Qualitative Research,* London: Chapman and Hall, pp. 466-480.

Walsham, G. (2006). Doing interpretive research, *European Journal of Information Systems* 15: 320-330.

Ward, R. P., Fayad, M. E. and Laitinen, M. (2001). Software Process Improvement in the Small - A Small Software Development Company's Most Difficult Challenge: Changing Processes to Match Changing Circumstances, *Communications of the ACM* 44(4): 105-107.

Wilkie, F. G., McFall, D. and McCaffery, F. (2005). An Evaluation of CMMI Process Areas for Small- to Medium-Sized Software Development Organizations, *Software Process: Improvement and Practice* 10(2): 189-201.

Appendix: Research papers

This appendix includes the four research papers that form the basis of this thesis.

Papers 1 (Hansen et al., 2004a, section 5.2), 2 (Tjornehoj and Mathiassen, 2008a, section 5.3) and 4 (Nielsen and Tjørnehøj, 2009) are all reproduced in this appendix as published. Paper 3 (Tjornehoj and Mathiassen, submitted 2007, section 5.4) is included as submitted for Journal of Information Systems, spring 2009. This paper was earlier published as a book chapter (Tjornehoj and Mathiassen, 2008b).

Appendix A:

Hansen, B., Rose, J. and Tjørnehøj, G. (2004a). Prescription, description, reflection: the shape of the software process improvement field, *International Journal of Information Management* 24(6): 457-472.

Appendix B:

Tjornehoj, G. and Mathiassen, L. (2008). Between control and drift: negotiating improvement in a small software firm, *Information Technology & People* 21(1): 69-90.

Appendix C: Tjornehoj, G. and Mathiassen, L. (submitted 2007). Improvisation during Process-technology Adoption: A Longitudinal Study of a Software Firm, *Journal of Information Technology*.

Appendix D:

Nielsen, P. A. and Tjornehoj, G. (2009). Social Networks in Software Process Improvement, *Software Process: Improvement and Practice*.

A. Prescription, description, reflection: the shape of the software process improvement field

Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
International Journal of Information Management 24 (2004) 457–472
International Journal of Information Management
www.elsevier.com/locate/ijinfomgt

Prescription, description, reflection: the shape of the software process improvement field

Bo Hansen[a,*], Jeremy Rose[b], Gitte Tjørnehøj[b]

[a] *Department of Informatics, Copenhagen Business School, Howitzvej 60 6, DK-2000 Frederiksberg, Denmark*
[b] *Department of Computer Science, Aalborg University, Fredrik Bajers Vej 7, DK-9220 Aalborg, Denmark*

Abstract

This article reviews 322 representative contributions to the software process improvement (SPI) literature. The contributions are categorised according to a simple framework: whether their primary goal is prescriptive (to tell SPI professionals what to do), descriptive (to report actual instances of SPI programs in software organisations), or reflective (theoretically analytical). The field is found to be rather dominated by one approach (the capability maturity model (CMM)) and heavily biased towards prescriptive contributions. Neither of these trends is necessarily beneficial, and it is argued that more theoretically reflective contributions could encourage a diversity of approaches which might also benefit practitioners.

Keywords: Software process improvement; Capability maturity model; Literature review

1. Introduction

Software process improvement (SPI) is an applied academic field rooted in the software engineering and information systems disciplines. It deals primarily with the professional management of software firms, and the improvement of their practice, displaying a managerial focus rather than dealing directly with the techniques that are used to write software. To date, it has been primarily practised and studied in America, Scandinavia and Australia. In terms of its theoretical heritage, SPI is equally indebted to the software engineering tradition and the Total

*Corresponding author. Tel.: +45-38-15-26-49.
E-mail addresses: bo@cbs.dk (B. Hansen), jeremy@cs.auc.dk (J. Rose), gtj@cs.auc.dk (G. Tjørnehøj).

0268-4012/$ - see front matter
doi:10.1016/j.ijinfomgt.2004.08.007

Quality Management movement (Deming, 1982; Juran & Gryna, 1988). Classical SPI techniques (such as those built upon the capability maturity model (CMM)) relate software processes, standardisation, software metrics and process improvement. However, the field has also expanded to include other approaches (such as the software factory approach) and (at first sight unrelated) issues such as the personal discipline of software engineers and commitment. SPI stakeholders include SPI practitioners (who are responsible for improvement programs), software supplier organisations and the organisations they contract for, government bodies sponsoring research, academics and consultants.

Many of the major contributions to SPI originate from the Software Engineering Institute (SEI) at Carnegie Mellon University (where Watts Humphreys has played a major innovative role). The Institute is industry-facing and supported by the American Department of Defense, whose principle interests are to identify competent software suppliers and ensure the delivery of high quality software. Analysis of the SEI's income for 2002 (the latest available year) showed that 65–73% of their income came from American Department of Defense or American military sources.[1] Many consultancy, teaching and licensing activities are also associated with the SEI, and their directly-sponsored project work amounted to half their income.

In this article, we develop a picture of the shape of the SPI field by analyzing it against three categories representing forms of writing. *Prescriptive* contributions are primarily concerned with informing SPI pratitioners how they can carry out software process improvement initiatives. *Descriptive* contributions are primarily concerned with describing those initiatives. *Reflective* contributions are primarily concerned with setting the other contributions in a theoretical context, or developing theory. The analysis framework is described more fully in Section 2.2. In principle this simple framework could be used to analyse any applied academic field. By developing such a picture, we expose some strengths and weaknesses of the field and contribute to focusing the direction of future research.

2. Research method

Webster and Watson (2002) suggest that literature reviews are an important part of the development of the IS field. They offer the opportunity to synthesize and reflect on previous theoretical work, thus providing secure grounding for the advancement of knowledge. They suggest that the elements of a good literature review include a structured approach to identifying the source material and the use of a concept matrix or other analytical framework leading to 'a coherent conceptual structuring of the topic'.

2.1. Article selection approach

The article selection approach focuses on identifying SPI related contributions from top IS journals, SPI heavy journals, special issues on SPI, literature review articles from within the SPI field, key SPI contributors, SPI schools (such as that based at SEI), key authors, relevant e-search tools, and finally identifying books written on SPI. To qualify, contributions should name software process improvement in the title, abstract or keywords and in addition have relevant

[1] http://www.sei.cmu.edu/publications/documents/02.reports/02ar/staff/funding-support.htm

content. Academic conference proceedings were included (but not prioritised) but non-academic sources such as practitioner journals and practitioner conferences were excluded. The approach is iterative, meaning that new finds lead to further improvement of the search criteria. Journals targeted include: *IEEE* (47 articles), *ACM* (15), *Communications of the ACM* (9), *MISQ* (2), *the DATABASE for Advances in Information Systems* (2), *the Journal of Systems and Software* (17), *Journal of Systems Architecture* (6), *Information and Software Technology* (5), *American Programmer* (4), *Journal of Knowledge Management* (1), *Journal of Software Process—Improvement and Practice* (15), and *Software Quality Journal* (20). At present, the resulting Endnote database contains 365 relevant contributions. We have over 150 contributions available in full-text format, and had enough additional information (in the form of abstracts, keywords, and notes) to categorize 322 entries. We expect to make the library (together with the categorization) available via ISWorld in due course for inspection. The library was originally compiled from the personal libraries of two Danish researchers (see acknowledgements) and for this reason still has a Scandinavian bias.

2.2. Categorisation framework: prescriptive, descriptive, reflective contributions

Here we develop the simple analysis framework that will be used to categorise the SPI field (Fig. 1). In principle, this framework could be used to analyse any applied academic field.

The first two categories are taken from Mintzberg's extensive survey of the strategy formation literature. He labels some of the strategy schools as *prescriptive*—'more concerned with how the strategies should be formulated than with how they necessarily *do* form.' (Mintzberg, 1990). Other kinds of schools are labeled *descriptive*—they are concerned 'less with prescribing ideal strategic behaviour than with describing how strategies do, in fact, get made' (Mintzberg, 1990). These categories can easily be related to the software process improvement literature, which is concerned both with specifying how software processes could or should be improved, and describing experiences of such improvement programs in software organisations. Since neither of these literature types are necessarily primarily concerned with the ongoing creation of secure theoretical knowledge (though they are not wholly divorced from it either), we add a third: *reflective*. Here we are less concerned with reflection in practice (Schon, 1983), than with what it means to be reflective in the context of an academic field. Reflection in this context will normally have an explicitly theoretical focus, using theory for analysis, or generating new theories or theoretical understandings. Reflection may be concerned with reviewing or categorizing prescriptions and/or descriptions against some form of theoretical canvas, or with generating such a canvas against which contributions to the field could better be understood. It may also be concerned with exposing or challenging basic taken-for-granted assumptions (the dominant paradigm (Kuhn, 1962)), thus more with double-loop than single-loop learning. It may focus on what Schein

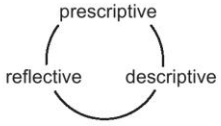

Fig. 1. Categorization framework for applied fields of academic study.

(Schein, 1973) calls the 'underlying discipline' or 'basic science' component upon which the engineering practice is based. Mintzberg's review of the strategy literature, for instance, could be classified as reflective in this academic context, in that it seeks to develop a framework (the ten strategy schools) against which many theoretical and practical experiences can be evaluated.

Using the terms prescriptive and (theoretically oriented) reflective, the evolution of an applied academic field could be represented as a cycle, in which

1. The (theoretically derived) prescriptions about practice are carried out in work situations.
2. The resulting experiences are precisely described in order to generate better understandings.
3. The resulting understandings are reflected over in order to generalise them to theory, which could then form the basis for better prescriptions.

3. Prescriptive SPI

The prescriptive contributions to SPI fall primarily into two categories: *norm-driven* and *problem-driven*. Norm-driven approaches (Arent, 2000; Aaen, Arendt, Mathiassen, & Ngwenyama, 2001) are based on an underlying normative model of software process improvement (which usually includes an explicit or implied normative model of software development—the processes to be improved); the main purpose for a SPI initiative is to align the software firm with this underlying model. By contrast, problem-driven approaches (Iversen, Nielsen, & Nørbjerg, 1999) prescribe how a software organisation can improve its problem-identification and -solving activities, and thus become better at identifying which parts of the development process need to be improved, and how to address this task.

3.1. Norm-driven approaches

Norm-based approaches to SPI display a common set of characteristics. They focus on software development processes at the organisational, project, team, or individual level, and are concerned with standardizing and improving those processes. They prescribe norms for how individuals, teams or organisations should operate, and for how processes should be standardised and improved. They assume that processes can be measured, both as a baseline for improvement and to provide indications of subsequent improvements. They normally assume that there are well-understood software development processes that everyone agrees can be recommended in all situations. Organisational improvement is normally related to a maturity ideal: the mature organisation has articulated, standardised, measurable software development processes (relating to normally inexplicit software engineering paradigm development ideals) and then measures in order to learn how to improve them further. Maturity levels can be measured, using various questionnaire-based techniques, and 'immature' organisations should normally follow a prescribed roadmap to achieve the next maturity level.

The CMM originating from the SEI at Carnegie Mellon University is probably the best known and most widely used approch to SPI. CMM is formally defined as 'a description of stages through which software organisations evolve as they define, implement, measure, control and improve their software process' (Paulk, Weber, Curtis, & Chrissis., 1995). The development of the

model was based on Watts Humphrey's characterization of the software process (Humphrey, 1988). The SEI worked with industry and government for 4 years to develop the CMM before publishing (Paulk, Curtis, Chrissis, & Weber, 1993), and it is still evolving, partly in response to feedback from the practitioners using it. The CMM principle has also been extended to other areas (Konrad, et al., 1996), including the Software Acquisition CMM, System engineering CMM, Integrated Product Management CMM, and People CMM. In 1997 the proliferation of models led to an effort to integrate them into CMM Integrated (CMMI) (Ahern, Clouse, & Turner, 2001; Reifer, 2002).

The SEI has also developed other norm-driven SPI approaches which are complementary to CMM. IDEAL is a practical approach to managing a software process improvement initiative developed at Carnegie Mellon. The activities (initiating, diagnosing, establishing, acting and leveraging) function as a normative managerial superstructure to a SPI iniative. The personal software process (PSP) (Humphrey, 1995) focuses on the individual discipline of the software engineer, in relation to the organisation's progression to maturity via the CMM. The emphasis is upon formal process, measurement, documentation, statistical assessment, scheduling and assessment techniques. Since writing professional programs is normally a team effort, rather than an individual effort, Humphreys also develops the team software process (TSP) (Humphrey, 1997, 1998, 2002). PSP and TSP could be seen as a response to early criticism that CMM was too process orientated and ignored the human factor: the contribution of professional software developers.

There are also normative SPI approaches developed outside SEI, of which the most prominent are European. The BOOTSTRAP methodology (Kuvaja & Bicego, 1994; Kuvaja, Similä, Krzanik, Bicego, Saukkonen, & Koch, 1994) was initially developed in an ESPRIT[2] project and is now the responsibility of the BOOTSTRAP Institute. Version 3.0 was released in September 1997 (Kuvaja, 1999). It combines elements of CMM with the relevant ISO, Department of Defense and European Space Agency software standards, in order to provide tools (essentially a questionnaire) for carrying out maturity assessments and thereafter making appropriate action plans. In TAPISTRY—a software process improvement approach tailored for small enterprises' (Kuvaja, Palo, & Bicego, 1999), the authors address the European market situation, which is characterized by many small and medium size enterprises that either cannot afford, or are not culturally suited to the full-scale assessment methods. The SPICE Project (Software Process Improvement and Capability dEtermination)[3] is the name commonly used for a project with the purpose of developing a working draft for a standard for software process assessment, conducting industry trials on this, and promoting the software process assessment to industry. The work originated from the existing assessment models and tried to develop a common base, on which the standard should rest. The SPICE standard does not in itself specify an assessment model or method, but defines a set of requirements that a model or method need to meet to comply with the standard.

3.2. Problem-driven approaches

Problem-driven approaches to SPI share some of the characteristics of the norm-based approaches, but are distinguished by focusing on ways to identify and solve specific problems in a

[2]http://www.cordis.lu/esprit/home.html
[3]http://www.isospice.com/spice/spiceproject.htm

Appendix A

software organisation (instead of prescribing a desirable model for developing software). They describe ways to improve an organisation's current practice and thus have a higher degree of freedom than the norm-driven approaches. They do not, therefore, normally incorporate assessments.

The problem-driven approaches covers a range of similar approaches to SPI (Cusumano, 1989; Aaen, Bøttcher, & Mathiassen, 1998) inspired by improvement efforts in manufacturing industry. The common assumption is that the software development process can be like the production process for industrial goods. As in the norm-driven approaches, software development is considered a repeatable process, consisting of sub-processes and procedures, which can be described to a certain level of detail. Systematization and standardization are the main coordination and formalization mechanisms, and the focus is on long-term, integrated, organisation-wide efforts which are centrally planned and managed.

The Industrialized Software Organisation or Japanese Software Factory Approach is not a formally defined approach, but more a practice evolved in large Japanese Co-operations. One example is Toshiba (Matsumoto, 1981, 1987) which operates a three phase model (design buildings supporting the development process, construct in integrated software support for the development process, and establish an monitoring and controlling system for the development process). By applying these steps, simple and repeatable work routines are created, which aim at heightening product quality without jeopardising the productive momentum.

The application of metrics in industry (AMI) project is the result of an ESPRIT funded project with the stated purpose of filling the gap between software process assessments and actual planning actions. The resulting method, described in an AMI handbook (Pulford, Kuntzmann-Combelles, Shirlaw, & Harutunian, 1992), is a pragmatic, incremental, quantitative approach applicable in any software business because of its claimed flexibility and adaptability to any organisation structure or team size.

The Generic Software Factory or the Eureka software factory project (ESF) (Fernström, 1991; Weber, 1997) was set up with the purpose of providing a generic architecture, a framework and to some extent a technological infrastructure for developing software factories, making it both easier for companies to build their own software factory and for companies to design tools supporting them in that.

With the experience factory, Basili, Caldiera, and Rombach (1994a) and Basili and Caldiera (1995) describe a somewhat different approach to problem-driven SPI in that they describe an infrastructure outlining a two-tier organisational structure; the *development organisation* and the *experience factory*. The first develops and delivers software and while doing this it also provides information to the latter. Based on this information the experience factory actively supports the development projects, and provides goals and models generated on the experiences drawn from previous experiences collected. A methodological support device, the quality improvement paradigm, is provided, consisting of a six-step cycle (understanding the process and product, definition of the process and product qualities, evaluation of successes and failures, information for project control, learning from experience, reusing of experience). By following these steps, the company can continuously learn from experience on two levels; the individual projects (the development organisation), and on the corporate level (the experience factory). An important tool in the experience factory is the goal/question/metric paradigm (Basili et al., 1994a) which supports the goal setting and measurement process.

4. Descriptive SPI

We report under this heading on three different categories of contributions that take as their principal focus, the reporting of actual SPI initiatives in companies. Much of the descriptive work experience relates with the CMM. The first category—we called the success stories—are reports of successful projects written by people heavily involved in the projects, such as CMM consultants, SPI project managers and staff of the SEI at Carnegie Mellon. Lack of credibility in the success stories caused researchers (led by staff at SEI) to look for more reliable methods of establishing the benefit of SPI initiatives, which we report under the heading of statistical surveys. There also exist a number of more independent, research oriented studies of SPI initiatives, and these we have labelled case studies.

4.1. Success stories

A strong element in the SPI literature concerns the narration of success stories. Examples are: NASA's Goddard Space Flight Centre (Basili, Caldiera, & Rombach, 1994b; Basili & Caldiera, 1995), Hughes Aircraft (Humphrey, Snyder, & Willis, 1991), Raytheon (Dion, 1992, 1993; Haley, 1996), PRC (Hollenbach, Young, Pflugrad, & Smith, 1997), Motorola (Diaz & Sligo, 1997), Oerlikon Aerospace (Laporte & Papicco, 1998), Shlumberger (Wohlwend & Rosenbaum, 1994). These are representative of the core American experience, principally with CMM, also with IDEAL and the experience-factory approach. Success stories tend to share most of the following characteristics:

- A generally positive tone about the SPI initiative.
- A large investment in the SPI program (presumably often including consultancy fees, though these are never reported).
- Authored by project participants—usually figures with responsibility for the SPI project, often members of SEI.
- Some evidence of success, either qualitative or quantitative, of a largely anecdotal nature (that is not collected or presented in a methodologically sound manner—which is not necessarily to say that it should not be believed).

Narration of success is combined with presentation of problems encountered lessons learned and advice for practitioners, which are, however, not generalised to theory. The problems described do not challenge the underlying paradigm, but relate more to the operationalisation of the prescribed approach (very often CMM) in the given context. Many of the well-reported success stories refer to relatively large, expensive projects in larger software firms connected to the American defence and aerospace industry.

Though largely descriptive in nature, these contributions are seldom descriptive in an objective, scholarly way and it is also normal that the principal point of the contribution is the lessons learned, which typically conclude the article: in other words, a prescriptive purpose. We therefore later chose to categorize many of these contributions as 'descriptive/prescriptive.'

4.2. Statistical surveys

Though the success stories provide interesting reading and serve to illustrate many of the practical hurdles in achieving SPI success, they lack 'representativeness' (Herbsleb & Goldenson, 1996). That is, they do not illustrate any general picture; especially where it is well known that many SPI initiatives are problematic or fail (Abrahamsson, 2002; Mathiassen, Pries-Heje, & Ngwenyama, 2002). The failures, however, are never reported. The SEI has made a fairly substantial effort to provide a more general, statistically based evaluation of their CMM experience (Herbsleb, Carleton, Rozum, Siegel, & Zubrow, 1994a; Herbsleb, Zubrow, Siegel, Rozum, & Carleton, 1994; Herbsleb & Goldenson, 1996). Part of this effort was a questionnaire sent to 167 CMM organisations. The respondents (83% of the sample) reported some statistically significant correlation between performance indicators (such as holding deadlines and budgets, and staff morale) and CMM level. Performance indicators reflect the perceptions of the (SPI-connected) respondents. However, few of the companies had achieved a high CMM level; according to the (un-assessed) informal judgment of the respondents themselves, 63% were at level 1 (the 'stuck in first' (Johnson & Brodman, 1996) phenomenon) and only 11% at level 3 or above. All the higher level organisations were 'government contractors'. When derived from the companies' (earlier) formal assessments, these figures were 83% (level 1) and 7% (level 3 or higher), respectively. Another study (Herbsleb et al., 1994a) showed substantial gains in productivity, defect detection and reduction time to market and business value (set against considerable investment). However, the sample involved 13 hand picked organisations, all large American, mainly government or defence related, three of which appear in the success story category, above (Herbsleb, Carleton, Rozum, Siegel, & Zubrow, 1994b).

Other contributors have made statistical surveys investigating very different subjects: CMM in small businesses (Brodman & Johnson, 1994; Bilotta & McGrew, 1998), the results and benefits of maturing (Goldenson & Herbsleb, 1995; Johnson & Brodman, 1996), the difficulty of examining ROI through CMM (Johnson & Brodman, 1995, 1996), and what are the success factors of CMM (El-Emam, Goldenson, McCurley, & Herbsleb, 2001) These surveys have drawn on data between 10 and 200 companies.

4.3. Case studies

In Scandinavia, researchers have carried out a number of case studies using some form of theoretical framework. Many of the SPI-programs involved were inspired by CMM. A metrics program in a large Danish company has been reported and reflected upon in a series of contributions. (Iversen, 2000; Frederiksen & Mathiassen, 2002; Frederiksen & Rose, 2003; Iversen & Mathiassen, 2003). Other research has focused on knowledge management and organisational learning (Arent & Nørbjerg, 2000; Kautz & Thaysen, 2001), on SPI in small companies (Kautz, Hansen, & Thaysen, 2000; Kautz, Thaysen, & Vendelø, 2002), on the personal software process (Abrahamsson & Kautz, 2002a,b), on commitment (Abrahamsson, 2001) and on a reflective usage of the IDEAL model (Borjesson & Mathiassen, 2003). These themes also figure in case studies from other parts of the world: metrics (Bhandari, Halliday, Tarver, Brown, Chaar, & Chillarege, 1993; Herbsleb & Grinter, 1998), knowledge management and learning (Gasston & Halloran, 1999; Conradi & Dingsoyr, 2000; Larsson & Kolb, 2002), small organisations (Kelly & Culleton,

1999). Other case studies touch on different subjects such as quality and SPI (Edgar-Nevill, 1994), managing diversity (Deck, 2001) and SPI in web time (Wiegers, 1999).

Though these contributions were primarily descriptive, the theroretical framework introduced a reflective element that lead us to later categorize many of them as 'descriptive/reflective'.

5. Reflective SPI

The reflective literature is sparse and differs much in style and purpose. Discussions range from on the core assumptions of CMM to the building of theoretical frameworks. The earlier contributions are focused on CMM itself, while the later tend to have a broader view of the SPI-field. A couple of early articles take a critical look at CMM (Bach, 1994) and CMM assessments (Bollinger & McGowan, 1991; Bach, 1994). The main criticisms are that CMM has no formal theoretical basis, and little empirical support, that it ignores people, reverses the institutionalisation of process for its own sake, and that it introduces an artificial goal (achieving a higher CMM level) in place of the goal of writing better software. The SEI (Curtis, 1994; Campbell, 1995) reply that the misconceptions are due to ignorance of CMM, point to CMM's reliance on the principles of total quality management, and suggests that the focus on process is justified by software development demanding a shared effort. Weaknesses in the grading templates and sparse data analysis are discussed, it is proposed that the grading system be abandoned and these themes continue to be discussed for some years (Bach, 1995; Fayad & Laitinen, 1997). Other strands of the reflective literature compare CMM and other approaches (Kohutek, 1996; Lyytinen, Mathiassen, & Ropponen, 1998) or analyse and discuss CMM from some theoretical standpoint. Ngwenyama & Nielsen (2003), for example, investigate the underlying values of CMM, revealing contradictory assumptions about organisational culture. Some later reflective contributions try to build frameworks of SPI either to characterize and define the field (Aaen et al., 2001) or to provide a tool for evaluation of different Software Process Models. (Hossein & Kalyani, 1999). Aaen et al. (2001) build a conceptual MAP based on an extensive survey of the SPI literature and experience from SPI practice, in which characteristic features (management, approach and perspective) of SPI are described. Hossein and Kalyani's evaluation framework focuses on goals of the model, structure, management role, use of metrics, benefits, underlying models, rating process, organisational impact and scope/domain.

6. Analysis

Simple analysis of the literature database first shows that the field is heavily influenced by the original CMM model. Searching for 'CMM' or 'Capability Maturity Model' in the title, abstract or keywords produced the following result (Fig. 2):

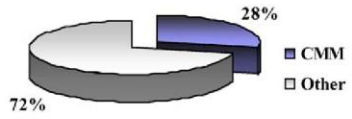

Fig. 2. The shape of the SPI literature, CMM versus other contributions.

Prescriptive P	136
Descriptive/ D/P	41
Prescriptive	
Descriptive D	74
Descriptive/ D/R	8
Reflective	
Reflective R	20
Total	322

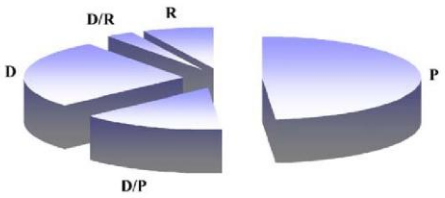

Fig. 3. The shape of the SPI literature: prescriptive, descriptive and reflective contributions.

However, many of the other contributions were also inspired by CMM, or one of the other SEI contributions. Next, the contributions were categorised against the prescription, description, reflection framework. It is understood that most of the contributions contain a mixture of these components, so the aim was to determine what the *primary* purpose of the contribution was. There were 322 items in the database with enough information to categorise. Nevertheless, some contributions proved hard to assign to the original framework and we devised two further categories to help us be fair to the nature of the contribution:

- Descriptive/prescriptive contributions describe an SPI initiative and conclude with prescriptive recommendations to other practitioners. This is the 'success story with lessons learned' contribution type, which very largely refers to CMM initiatives. However, there is no theoretical background apart from CMM itself.
- Descriptive/reflective contributions principally describe a SPI initiative but conclude with reflective work (according to our academic definition of reflection). These mainly refer to the case study type of contribution.

The results of this analysis are given in Fig. 3.

We therefore conclude that the SPI literature in our sample is weighted heavily towards, prescriptive contributions, and shows less evidence of academically reflective work. Some implications of this conclusion are discussed in the next section.

7. Discussion

This section discusses two features of the SPI literature, the first related to the dominant role of the CMM model, and the second, related to the overall shape of the field.

7.1. The CMM refinement learning circle

In a field heavily dominated by one product (the original CMM model), one form of learning cycle is particularly evident (Fig. 4). The cycle reflects the SEI's industry orientation and close co-operation with industry. The prescriptions of CMM are fairly extensively tried out in industrial settings, and the practitioners and consultants involved report back with refinements or adaptations to the CMM model, often relating to a CMM implementation in a specific context

Fig. 4. The CMM refinement learning cycle.

(small firm, different culture, etc.). This results in an expanding body of knowledge about CMM and how to use it, and the particular type (prescriptive/descriptive) of SPI literature in which SPI practitioners describe their project and draw lessons for the guidance of other SPI practitioners.

However, the principal question addressed in the 'lessons learned' contributions is 'how can we best implement CMM?' That is to say that the cycle only really addresses single loop learning and the basic assumptions of the CMM model are never questioned. All of the examples in this genre (and of course the 'success story' genre) report essentially successful CMM projects. 'Success' is largely defined as progress up the CMM levels, whilst consequent benefit to the company's ability to write useful software and to return a profit is left un-examined. Statistical surveys at SEI try (rather unconvincingly) to address this problem, by linking expenditure on CMM projects to efficiency savings and expanding earnings. If this literature is taken at face value, CMM is an extremely successful phenomenom. However, it may be that the characteristics of these reports (consultant and practitioner authors, heavy investment in CMM projects, a product (CMM) that must be sold, commercial companies that must advertise themselves in a market, powerful government sponsors) mean that only the success stories are told. Who wants to advertise that they invested a lot of time, money and energy in a project that was a flop? Some of the Scandinavian research experience, and reading between the lines of SEI's own figures, however suggests that there may be many companies that start a CMM program, make little progress up the hierarchy, and later abandon the project.

Refinement or learning circles like the one described above are extremely effective for making research ideas work in practical situations. It seems clear that, at least in one particular setting (large American software companies writing software for defence contractors), CMM has been an extremely successful improvement tool. However, we speculate that this may be as much due to the extremely close financial relationships between research sponsors, researchers, software vendors and software buyers, as to the quality of the improvement tool. It is much less clear whether CMM has successfully transferred to other environments and cultures. Furthermore, this CMM learning cycle represents a single loop learning cycle, in as much as the basic assumptions of the original model are hardly ever questioned. However, there is evidence of more reflective learning in other aspects of SEI's work, such as the development of continuous (non level based) process improvement and the people maturity model.

7.2. Prescriptive (non-reflective) academic fields

In our analysis, at least, the SPI field appears somewhat overly prescriptive and lacking in reflective contributions. It may be that this trend has also contributed to the dominance of CMM. The comparative lack of critical scrutiny, rigorous descriptive research carried out by trained neutral researchers, and the theoretical influence from other related fields, together with the focus on applied techniques rather than the building of defensible theory, has meant that credible

alternative ways of improving the making of software in software firms have either not yet emerged, or not been able to compete.

However, we have not performed similar analysis of other applied fields, and know of no similar analysis, so we are not able to conclude how typical the shape of the SPI field is. Certainly, there are many prescriptive contributions to the related disciplines of software engineering, information systems, and management. The prescriptive genres may be prevalent in engineering disciplines (this may be one explanation); however, it is not clear that the improvement of the building of software is primarily an engineering task—it could equally be a managerial task. Nevertheless, we take a provocative guess that the SPI field has a less reflective character than many other fields. In this respect, it may (again provocatively) be described (in the language of CMM) as 'immature.' This may not matter much if the techniques are anyway effective in industry, but we have some difficulty establishing, partly because of the unreflective nature of the field, whether this is actually the case.

Many of the more reflective contributions come from the Scandinavian school. This may also point to a Scandinavian bias in the whole analysis, in as much as the reflective elements count more (also to these researchers) in some traditions of research than others. Practical successes may count more in other traditions. Therefore it may be that we effectively evaluate one tradition of research against the standards of a different tradition—not necessarily a productive thing to do. However, we would be very hesitant to characterize American research in management or information systems as non-reflective—this seems absurd.

We conclude that there may be some good reasons for trying to develop the more reflective, theoretically oriented strand of SPI research.

8. Conclusions

In this paper, we assessed the shape of the SPI field by categorizing a substantial number of contributions to its academic literature as prescriptive, descriptive or reflective. Though there could be many objections to the way we built the database, the analysis framework, and to the way we categorised the contributions, we think our two main conclusions would hold under any analysis of the field. We conclude that the field is rather dominated by CMM, and that it is a rather prescriptive and non-reflective field. Neither of these trends is necessarily beneficial to the field as a whole. Whilst acknowledging the very many successes and innovations of the SEI with CMM, it has never been clear that it is widely appropriate or successful outside its natural habitat (though it is very widely known), and its very success, coupled with the lack of serious reflective challenges to it, may have stifled the development of a more multi-faceted range of approaches (such as exist in the systems development field). Such a multi-faceted range of approaches could well be beneficial in the wide range of cultural and situationally different circumstances under which the software is built.

In order to build on the existing reflective work in the SPI, we would encourage the following types of SPI research:

- Descriptive studies of SPI initiatives carried out by trained independent researchers.
- Statistical studies across SPI projects with sound methodological foundations carried out by independent researchers.

- Theoretical analysis of such descriptions using theories from related disciplines.
- Reflective cumulative accounts of trends.
- The building of theoretical accounts of improvements in software construction based on relevant theories, and/or independently researched descriptions of actual projects.

Such forms of research may redress the balance a little, and ensure a better balance between theory and practice in the future.

Acknowledgements

The research was in part made possible by the Danish Government through their sponsorship of the Software Processes and Knowledge project.

Our thanks to Peter Axel Nielsen and Ivan Aaen for providing the original EndNote libraries.

References

Aaen, I., Arendt, J., Mathiassen, L., & Ngwenyama, O., 2001. A conceptual MAP of software process improvement. Scandinavian Journal of Information Systems, 13.

Aaen, I., Bøttcher, P., & Mathiassen, L. (1998). *The Software Factory: Contributions and Illusions. Sixth European Conference on Information Systems (ECIS)*. France: Aix en Provence.

Abrahamsson, P. (2001). Rethinking the concept of commitment in software process improvement. *Scandinavian Journal of Information Systems*, *13*, 69–98.

Abrahamsson, P. (2002). Commitment nets in software process improvement. *Annals of Software Engineering*, *14*(1–4), 407–438.

Abrahamsson, P., & Kautz, K. (2002a). *Personal software process—classroom experiences from Finland. Proceedings of the seventh European conference on software quality*. Finland: Helsinki.

Abrahamsson, P., & Kautz, K. (2002b). *The personal software process—experiences from Denmark. Proceedings of the 28th EUROMICRO 2002 conference*. Germany: Dortmund.

Ahern, D. M., Clouse, A., & Turner, R. (2001). *CMMI Distilled: An Introduction to Multi-discipline Process Improvement*. Reading, MA: Addison-Wesley.

Arent, J. (2000). Normative software process improvement. Department of Computer Science, Aalborg University.

Arent, J., & Nørbjerg, J. (2000). *Software process improvement as organisational knowledge creation—a multiple case analysis. Proceedings of the 33rd Hawaii international conference on system sciences*. Hawaii: Wailea.

Bach, J. (1994). The immaturity of the CMM. *American Programmer*, *7*(9), 13–18.

Bach, J. (1995). Enough about process: what we need are heroes. *IEEE Software*, *12*(2), 96–98.

Basili, V., Caldiera, G. (1995). Technical Report: CS-TR-3483, UMIACS-TR-95-67, Univeristy of Maryland, College Park, MD 20742, USA.

Basili, V., Caldiera, G., & Rombach, H. D. (1994a). *The experience factory. Encyclopedia of software engineering—2 Volume Set*. New York: Wiley.

Basili, V., Caldiera, G., & Rombach, H. D. (1994b). *The goal question metric approach. Encyclopedia of software engineering—2 volume set*. New York: Wiley p. 528–532.

Bhandari, I., Halliday, M., Tarver, E., Brown, D., Chaar, J., & Chillarege, R. (1993). A case-study of software process improvement during development. *IEEE Transactions on Software Engineering*, *19*(12), 1157–1170.

Bilotta, J. G., & McGrew, J. F. (1998). A Guttman scaling of CMM level 2 practices: investigating the implementation sequences underlying software engineering maturity. *Empirical Software Engineering,* (3), 159–177.

Bollinger, T. B., & McGowan, C. (1991). A critical look at software capability evaluations. *IEEE Software*, *8*(4), 25–41.

Appendix A

Borjesson, A., & Mathiassen, L. (2003). Making SPI happen: the IDEAL distribution of effort. System Sciences. Proceedings of the 36th annual Hawaii international conference . (pp. 328–337).

Brodman, J. G., & Johnson, D. L. (1994). What small businesses and small organizations say about the CMM. *Proceedings of the 16th international conference on software engineering*, Sorrento, Italy.

Campbell, M. (1995). *Tool support for software process improvement and capability determination: Changing the paradigm of assessment. Software process newsletter*. Silver Spring, MD: IEEE Computer Society (No. 4) p. 12–15.

Conradi, R., & Dingsoyr, T. (2000). Software experience bases: a consolidated evaluation and status report. *Product Focused Software Process Improvement*, *1840*, 391–406.

Curtis, B. (1994). A mature view of the CMM. *American Programmer*, *7*(9), 19–27.

Cusumano, M. A. (1989). The software factory: a historical interpretation. IEEE Software.

Deck, M. (2001). Managing process diversity while improving your practices. *IEEE Software*, *18*(3), 21–27.

Deming, W. E. (1982). *Out of the crisis*. Cambridge, MA: Productivity Press.

Diaz, M., & Sligo, J. (1997). How software process improvement helped Motorola. *IEEE Software*, *14*(5), 75–81.

Dion, R. (1992). Elements of a process-improvement program. *IEEE Software*, *9*(4), 83–85.

Dion, R. (1993). Process improvement and the corporate balance sheet. *IEEE Software*, *10*(4), 28–35.

Edgar-Nevill, V. M. A. (1994). Evaluation of the SEI software capability model within an information systems context; in pursuit of software quality. In: Ross, M., Brebbia, C. A., Staples, G., Stapleton, J. (Eds.). Software quality management II. Managing quality systems (Vol. 1, pp. 263–278). Computer Mechanical Publications.

El-Emam, K., Goldenson, D., McCurley, J., & Herbsleb, J. (2001). Modelling the likelihood of software process improvement: an exploratory study. *Empirical Software Engineering*, *6*(3), 207–229.

Fayad, M. E., & Laitinen, M. (1997). Process assessment considered wasteful. *Communications of the ACM*, *40*(11), 125–128.

Fernström, C. (1991). The Eureka software factory, concepts and accomplishments. *Third European software engineering conference*. Berlin: Springer.

Frederiksen, H. D., & Mathiassen, L. (2002). Diagnosing metrics practice in a software organisation. In Harindranath, G., Rosenberg, D., & Sillince, J. A. A., et al. (Eds.), *New perspectives on information systems development, theory, methods and practice*. New York: Kluwer Academic.

Frederiksen, H. D., & Rose, J. (2003). The social construction of the software operation: reinforcing effects in metrics programs. *Scandinavian Journal of Information Systems*, *15*, 23–38.

Gasston, J., & Halloran, P. (1999). Continuous software process improvement requires organisational learning: an Australian case study. *Software Quality Journal*, *8*(1), 37–51.

Goldenson, D., & Herbsleb, J. D. (1995). *After the appraisal: a systematic survey of process improvement, its benefits, and factors that influence success*. CMU/SEI-95-TR-009. Software Engineering Institute, Pittsburgh, PA.

Haley, T. J. (1996). Software process improvement at Raytheon. *IEEE Software*, *13*(6), 33–41.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994a). *Benefits of CMM-based software process improvement: executive summary of initial results.* SPECIAL REPORT CMU/SEI-94-SR-013, Carnegie Mellon University.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994b). *Benefits of CMM-based software process improvement: initial results*. CMU/SEI-94-TR-13, Software Engineering Institute, Pittsburgh, PA

Herbsleb, J. D., & Goldenson, D. R. (1996). A systematic survey of CMM experience and results. ICSE-18, IEEE.

Herbsleb, J. D., & Grinter, R. E. (1998). *Conceptual simplicity meets organisational complexity: Case-study of a corporate metrics program. Proceedings of the 1998 international conference on software engineering: forging new links*. Silver Spring, MD: IEEE Computing Society.

Herbsleb, J., Zubrow, D., Siegel, J., Rozum, J., & Carleton, A. (1994). Software process improvement: state of the payoff. *American Programmer*, *7*(9), 2–12.

Hollenbach, C., Young, R., Pflugrad, A., & Smith, D. (1997). Combining quality and software improvement. Association for computing machinery. *Communications of the ACM*, *40*(6), 41.

Hossein, S., & Kalyani, C. (1999). Towards an evaluative framework for software process improvement models. *The Journal of Systems and Software*, *47*(2,3), 139.

Humphrey, W. S. (1995). *A discipline for software engineering*. Reading, MA: Addison Wesley Publishing Company.

Humphrey, W. S. (1997). *Managing technical people—innovation, teamwork, and the software process*. Pittsburgh: Addison-Wesley.

Humphrey, W. S. (1988). Characterizing the software process. *IEEE Software*, *5*(2), 73–79.
Humphrey, W. S. (1998). Three dimensions of process improvement. Part I: process maturity. pp. 1–7.
Humphrey, W. S. (2002). Three process perspectives: organisations, teams, and people. *Annals of Software Engineering*, *14*(1–4), 39–72.
Humphrey, W. S., Snyder, T. R., & Willis, R. R. (1991). Software process improvement at Hughes aircraft. *IEEE Software*, *8*(4), 11–23.
Iversen, J. (2000). Data-driven intervention in software process improvement. Department of Computer Science, Aalborg University.
Iversen, J., & Mathiassen, L. (2003). Cultivation and engineering of a software metrics program. *Information Systems Journal*, *13*(1), 3–19.
Iversen, J., Nielsen, P. A., & Nørbjerg, J. (1999). Situated assessment of problems in software development. *The DATABASE for advances in information systems*, *30*(2).
Johnson, D., & Brodman, J. (1996). Realities and rewards of software process improvements. *IEEE Software*, *13*(6), 99–101.
Johnson, D. L., & Brodman, J. G. (1995). Return on investment (ROI) from software process improvement as measured by US industry. *Software process—improvement and practice*, *1*(1), 35–47.
Juran, J. M., & Gryna, F. (1988). *Juran's quality control handbook*. New York: McGraw-Hill Book Company.
Kautz, K., & Thaysen, K. (2001). Knowledge, learning and IT support in a small software company. *Journal of Knowledge Management*, *5*(4), 349–357.
Kautz, K. H., Hansen, H. W., & Thaysen, K. (2000). *Applying and adjusting a software process improvement model in practice: the use of the IDEAL model in a small software enterprise*. Limerick, Ireland: International Conference on Software Engineering.
Kautz, K. H., Thaysen, K., & Vendelø, M. T. (2002). Knowledge creation and IT systems in a small software firm. Journal of OR Insight.
Kelly, D. P., & Culleton, B. (1999). Process improvement for small organisations. *Computer*, *32*(10), 41.
Kohutek, H. J. (1996). Reflections on the capability and maturity models of engineering processes. *Quality and Reliability Engineering International*, *12*(3), 147–155.
Konrad, M., Chrissis, M. B., Ferguson, J., Garcia, S., Hefley, B., Kitson, D., & Paulk, M. (1996). Capability maturity modeling at the SEI. *Software Process—Improvement and Practice*, *2*(1), 21–34.
Kuhn, T. (1962). *The structure of scientific revolutions*. Chicago: Chicago University Press.
Kuvaja, P. (1999). BOOTSTRAP 3.0—A SPICE1 conformant software process assessment methodology. *Software Quality Journal* (8), 7–19.
Kuvaja, P., & Bicego, A. (1994). BOOTSTRAP—a European assessment methodology. *Software Quality Journal*, *3*(3), 117–127.
Kuvaja, P., Palo, J., & Bicego, A. (1999). TAPISTRY—a software process improvement approach tailored for small enterprises. *Software Quality Journal*, *8*(2), 149–156.
Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Saukkonen, S., & Koch, G. (1994). *Software process assessment & improvement—the bootstrap approach*. Oxford: Blackwell Publisher.
Laporte, C., & Papicco, N. (1998). Software and systems engineering process development and integration at Oerlikon aerospace. *Software Process Newsletter*, *11*, 10–17.
Larsson, S., & Kolb, P. (2002). Software process improvement at ABB. *Proceedings of the Institution of Civil Engineers-Civil Engineering*, *150*, 46–49.
Lyytinen, K., Mathiassen, L., & Ropponen, J. (1998). Attention shaping and software risk, a categorial analysis of four classical approaches. *Information Systems Research*, *9*(3).
Mathiassen, L., Pries-Heje, J., & Ngwenyama, O. (Eds.) (2002). *Improving software organisations: from principles to practice*. Reading, MA: Addison-Wesley.
Matsumoto, Y. (1981). *SWB System: A software factory*. Amsterdam: North-Holland.
Matsumoto, Y. (1987). *A software factory: an overall approach to software production*. London: IEEE.
Mintzberg, H. (1990). Strategy formation: schools of thought. In Fredrickson, J. W. (Ed.), *Perspectives on strategic management*. New York: Harper Business.
Ngwenyama, O., & Nielsen, P. A. (2003). Competing values in software process improvement: an assumption analysis of CMM from an organisational culture perspective. *IEEE Transactions on Engineering Management*, *50*(1), 100–112.

Appendix A

Paulk, M. C., Curtis, B., Chrissis, M. B., & Weber, C. (1993). *Capability maturity model for software version. 1.1*. Pittsburgh, PA: Software Engineering Institute.

Paulk, M. C., Weber, C., Curtis, B., & Chrissis, M. B. (1995). *The capability maturity model: guidelines for improving the software process*. Reading, MA: Addison Wesley.

Pulford, K., Kuntzmann-Combelles, A., Shirlaw, S., & Harutunian, K. (1992). *A quantitative approach to software management: the AMI handbook*. London: CSSE South Bank University.

Reifer, D. J. (Ed.). (2002). The CMMI: it's formidable [Guest editor's corner]. *The Journal of Systems and Software*, *50*, 97–98.

Schein, E. (1973). *Professional education*. New York: McGraw-Hill.

Schon, D. (1983). *The reflective practitioner: how professionals think in action*. New York: Basic Books.

Weber, H. (1997). *The software factory challenge*. Amsterdam: IOS Press.

Webster, J., & Watson, R. T. (2002). Editorial: analyzing the past to prepare for the future: writing a literature review. *MIS Quarterly*, *26*(2), xiii–xxiii.

Wiegers, K. (1999). Software process improvement in web time. *IEEE Software*, *16*(4), 78–86.

Wohlwend, H., & Rosenbaum, S. (1994). Schlumberger's software improvement program. *IEEE Transactions on Software Engineering*, *20*(11), 833–839.

Bo Hansen, M.Sc. (Computer Science and Business Administration), is Ph.D. Student at Department of Informatics, Copenhagen Business School, Denmark. His research interest are in knowledge management and software development practices, and he is involved in a national research project addressing Software Process Improvements (SPV project). Previously he worked as a systems engineer in Danish Internet and Investment companies.

Jeremy Rose is Associate Professor at the Department of Computing Science, Aalborg University, Denmark. He is currently a member of the PITNIT and SPV research projects in Denmark, and active as a member of the IFIP WG8.2 community. His research interests are principally concerned with IT and organisational change, the management of IT and IS development. The research approach combines empirical insights with theoretical perspectives to produce practically useful knowledge around various contemporary technologies in different types of organisational settings. He has published in management, systems and IS forums. Further details and some publications are available at http://www.cs.auc.dk/~jeremy/.

Gitte Tjørnehøj is assistant professor and Ph.D. student at the Department of Computing Science, Aalborg University, Denmark. She is a member of the SPV research project in Denmark. Her research is rooted in 10 years experience working in the IT-industry. Her main research interests are management of IT and IS development focused on organisational change.

B. Between control and drift: negotiating improvement in a small software firm


The current issue and full text archive of this journal is available at
www.emeraldinsight.com/0959-3845.htm

Between control and drift: negotiating improvement in a small software firm

Gitte Tjornehoj
Department of Computer Science, Aalborg University, Denmark, and
Lars Mathiassen
Computer Information Systems, J. Mack Robinson College, Georgia State University, Atlanta, Georgia, USA

Received 27 September 2006
Revised 21 March 2007
17 October 2007
Accepted 26 October 2007

Abstract

Purpose – While the literature on software process improvement (SPI) offers a number of studies of small software firms, little is known about how such initiatives evolve over time. On this backdrop, this paper aims to investigate how adoption of SPI technology was shaped over a ten year period (1996-2005) in a small Danish software firm.

Design/methodology/approach – The investigation is based on a longitudinal, interpretative case study of improvement efforts over a ten-year period. To help structure the investigation, we focus on encounters that impacted engineering, management, and improvement practices within the firm. The study contributes to the SPI-literature and the literature on organizational adoption of technology.

Findings – The paper finds the improvement effort fluctuating and shaped between management's attempt to control SPI technology adoption and events that caused the process to drift in unpredictable directions.

Practical implications – The experiences suggest that managers of small software firms remain flexible and constantly negotiate technology adoption practices between control and drift, creating momentum and direction according to firm goals through attempts to control, while at the same time exploring backtalk, options, and innovations from drifting forces inside and outside the firm.

Originality/value – Based on the research, the paper recommends substituting the "from control to drift" perspective on organizational adoption of complex technologies like SPI with a "negotiating control and drift" perspective.

Keywords Technology led strategy, Computer software, Control systems, Denmark

Paper type Research paper

Introduction

In his latest work, Claudio Ciborra addressed modernity and the consequences for managing technology in industry (Ciborra *et al.*, 2000; Ciborra, 2002). He argued that traditional thinking of successful technology adoption as planned and managed does not suffice in the modern, global world. On the contrary, technology drifts away from plans just as fast as new plans are made. What technology ends up being is a result of events in a complex, unpredictable, and unmanageable web that is shaped by bricolage

The authors dedicate this paper to Claudio Ciborra (1951-2005) for his contributions to Information Systems theory. Claudio thrived in controversy and he was particularly critical to CMM and much of the SPI literature. In that spirit, the authors have developed Claudio's thinking on technology adoption based on a detailed study of ten years of SPI practices in a small software firm.

Information Technology & People
Vol. 21 No. 1, 2008
pp. 69-90

0959-3845
DOI 10.1108/09593840810860333

(Ciborra, 1994a, Ciborra and Hanseth, 1998, Lévi-Strauss, 1962, 1972) and formative context (Ciborra and Lanzara, 1994). The world is moving, he argued, from being governed through control towards being dominated by side effects, drifting technologies, and increasing unpredictability and risk.

In this context of growing complexity, software process improvement (SPI) has become an increasingly interesting technology for software firms struggling to stay competitive. The Capability Maturity Model (CMM) (Paulk *et al.*, 1993) and the recent CMM Integrated (CMMI Product Development Team, 2002, CMMI product development Team, 2000) have sparked a new discipline of engineering management that plays a dominating role in practice and research (Hansen *et al.*, 2004). The CMMs are based on the ideal of a rational, control-centered culture for software development (Ngwenyama and Nielsen, 2003) and although other SPI approaches have been suggested (Ares *et al.*, 2000, Bennetts and Wood-Harper, 1998; Kehoe and Shah-Jarvis, 1995; Schneider, 2002) they do not differ from the CMMs when it comes to underlying values (Hansen *et al.*, 2004).

In general, SPI research is based on short periods of empirical evidence. While the literature offers a number of successful cases of adopting CMM (Humphrey *et al.*, 1991, Wohlwend and Rosenbaum, 1994, Dion, 1992), reports on failure and difficulties have lately increased (Mathiassen *et al.*, 2002; Ngwenyama and Nielsen, 2003; El-Emam *et al.*, 2001; Rodenbach *et al.*, 2000; Börjesson and Mathiassen, 2003; Villalon *et al.*, 2002; Börjesson and Mathiassen, 2005; Hansen *et al.*, 2004). Especially within small software firms, adoption of SPI-technology seems problematic since these organizations lack sufficient resources to invest in improvements (Steel, 2004; Brouse and Buys, 1999; Villalon *et al.*, 2002; Brodman and Johnson, 1994) and SPI knowledge is not sufficiently tailored to their needs (Kilpi, 1998; Saastamoinen and Tukiainen, 2004). Small software firms are highly sensitive to dynamic environments (Ward *et al.*, 2001) and dominating approaches to SPI fit poorly (Leung and Yuen, 2001; Varkoi *et al.*, 1999) because it normally takes several complex and expensive initiatives to reach new maturity levels (Aaen *et al.*, 2001).

On this basis, we have adopted Ciborra's framework to investigate the following research question "How can small software firms manage adoption of SPI-technology?" Our focus is hence not on specific maturity models or software processes, but on how adoption of SPI-technology (including maturity models, improvements approaches, principles for organization, and tools) can help small software firms improve engineering practices. The investigation is based on a longitudinal, interpretative case study (Walsham, 1993, Pettigrew, 1990) of improvement efforts over a ten-year period from 1996 to 2005 in a Danish software firm, SmallSoft. To help structure the investigation, we focus on encounters that impacted engineering, management, and improvement practices within the firm (Peterson, 1998; Cho *et al.*, 2006; Newman and Robey, 1992). The study contributes to the SPI-literature and the literature on organizational adoption of technology.

The argument is structured as follows. First, we present relevant theory on SPI in small firms and introduce the concepts of control and drift based on the latest work of Ciborra (Ciborra, 2002; Ciborra *et al.*, 2000). Then, we explain our longitudinal, interpretative case study approach (Walsham, 1993; Pettigrew, 1990) structured around key encounters (Cho *et al.*, 2006; Newman and Robey, 1992; Peterson, 1998). Subsequently, we introduce SmallSoft and provide a detailed account of how

SPI-technology was adopted during the period 1996-2005. Finally, we discuss the case in response to the research question and highlight the contributions and implications of the study.

Theoretical background

Small software firms face special challenges when improving software practices. These challenges relate to the resources available and to having minimal influence over the environment. In the following, we review what we know about SPI in small software firms and we present Ciborra's (2002) distinction between control and drift as a framework for investigating management challenges in this particular context.

Improvement in small software firms

There are a number of studies of small firms trying to adopt CMM. Typically, these studies describe the difficulties that small software firms encounter and how these can be successfully resolved. Several studies suggest adaptations of CMM to small firms ranging from planning and implementing SPI-initiatives pragmatically according to firm needs (Batista and Figueiredo, 2000; Harjumaa *et al.*, 2004; Varkoi *et al.*, 1999; Kilpi, 1998; Scott *et al.*, 2002; Jakobsen, 1998; Kautz *et al.*, 2000; Kautz, 1999; Kelly and Culleton, 1999) to composing new SPI-frameworks tailored for small firms and comprised from CMM key process areas (Leung and Yuen, 2001; Wilkie *et al.*, 2005, Casey and Richardson, 2004; Kautz *et al.*, 2000; Horvat *et al.*, 2000; Ruiz *et al.*, 2002). The main insight offered by this research is that adapting CMM can be beneficial to small firms when done through pragmatic interpretations of available frameworks. Managers are advised to consider in which sequence, with what focus, and on which level of ambition different CMM process areas are applied to assess current practices and implement new ones. In support of such a pragmatic approach, Dybå (2003) found that among 120 Scandinavian companies the smaller ones adopted SPI-technology just as effectively as larger companies. Also, Paulk (1998) argues that small firms' adoption of CMM "may be different in degree, but they are not different in kind" (Paulk, 1998) from those of other organizations. In general, it takes "professional judgment and understanding of how the CMM is structured to be used for different purposes" (Paulk, 1998).

Another group of studies have discarded CMM and developed alternative approaches while still keeping the basic values of the rational software organization intact. Most of these studies report from one or a few cases of successful application and, on that basis, recommend the presented approach to other small software firms. Examples are DSDM and people process (Coleman and Verbruggen, 1998), AMETIST (Thowart, 1999), TAPESTRY (Kuvaja *et al.*, 1999), 3P approach (Brouse and Buys, 1999), IMPACT (Scott *et al.*, 2001), Software Process Matrix (Richardson, 2001, 2002), MESOPYME (Villalon *et al.*, 2002), COSMEA (Grechenig and Zuser, 2004; Steel, 2004), and SPICE (Truffley *et al.*, 2004). While these studies vary considerably in focus, they all advocate a need for adopting SPI-technology and they propose ways to downsize the effort to better match the resources and culture of small software firms.

Managers in small software firms can hence find advice for SPI-technology adoption in the literature. Most of the studies are, however, based on observations over limited time periods, focusing on initial adoption. Only a few studies report on how SPI-initiatives evolve over a considerable time span (Balla *et al.*, 2001; Truffley *et al.*,

2004; Richardson, 2001; Kuvaja *et al.*, 1999; Mustonen-Ollila and Lyytinen, 2003). Moreover, most studies focus on adapting elements of CMM or on developing alternative models that fit the needs of small software firms. In this study, we have therefore pushed models into the background and given center stage to the understanding of how SPI-technology was adopted in a small software firm over a period of ten years. To make sense of this adoption process we focus on the relationship between managerial control and drift caused by events inside and outside the firm.

From control to drift

In the book *From Control to Drift* Claudio Ciborra *et al.* (2000) outline a vicious circle of how firms strive for management control of technology adoption, but instead experience drifting due to forces of turbulent environments, implementation tactics, power of the installed base, complexity of the technology, side-effects, surprises, and users' resistance and creativity. Limits to learning and the pre-existing formative context (Ciborra and Lanzara, 1994) keep firms in this cycle and reinforce the perceived need for control.

Ciborra and his associates focus on corporate information infrastructures as elements in globalization, but they also discuss technology adoption in general (Ciborra *et al.*, 2000). Ciborra (2002) elaborates further and points to a hidden crisis of the Information Systems field. This crisis is caused by the dominating role of rational models in managing organizations and technology. He argues that this view is constraining our understanding of the world and prevents us from seeing other dimensions of technology (Ciborra, 2002).

The rational science view leads us, according to Ciborra (2002), to misunderstand or not see the world as experienced in the everyday life of agents, users, designers, and managers. We introduce and enforce geometrical models trying to make the world fit. But while trying to control and plan technology in this way, it drifts away from plans as side effects (Hanseth *et al.*, 2001) and surprises (Ciborra, 1994b) happen. Humans respond by reinventing technology through improvisations, bricolage (Ciborra, 1994a, Ciborra and Hanseth, 1998, Lévi-Strauss, 1962, 1972), and hacking[1] (Ciborra, 1999). The adoption process is therefore shaped differently than expected through formative contexts (Ciborra and Lanzara, 1994) or the already installed base (Ciborra and Hanseth, 1998). When bounded in their imagination by specific formative contexts (Ciborra and Lanzara, 1994), humans have limited innovative capabilities. However, coincidence, breakdowns, and human coping can spark technology drifting and result in more innovative outcomes:

> Drifting can be looked at as the outcome of two intertwined processes. One is given by the openness of the technology, its plasticity in response to the re-inventions carried out by users and specialists, who gradually learn to discover and exploit features, affordances, and potentials of systems. On the other hand, there is the sheer unfolding of the actors' being in the work flow and the continuous stream of interventions, tinkering, and improvisations that color perceptions of the entire system life cycle (Ciborra, 2002, p. 87).

In this view, usage, maintenance, redevelopment, and improvement take place simultaneously and range from sabotage over passive resistance, to learning and micro inventions, and even radical shifts. As a result, Ciborra argues, we need to change our thinking and practices from control to drift. Such a move will allow firms to support

human innovation instead of controlling, and facilitate cultivating and hosting technology instead of planning and designing it. These ideas have recently been addressed and employed in the special issue of *Journal of Information Systems* on "Claudio Ciborra and the IS Field: Legacy and Development" (Brigham and Introna, 2006; Saccol and Reinhard, 2006; Elbanna, 2006).

Although the adoption of SPI-technology in SmallSoft differs from Ciborra and his associates' cases of corporate infrastructures of global companies (2000) in terms of company size, the interconnectedness, unpredictability, and degree of external pressure on the firms are similar. SPI-technology displays characteristics of openness and plasticity and is also promoted as having strategic importance to firms. Hence, the discourse outlined in Ciborra's later work lends itself nicely to making sense of the particular case of SPI-technology adoption.

Research approach

Longitudinal interpretative case study

We have framed the investigation as a longitudinal, interpretative case study (Walsham, 1993, 1995; Pettigrew, 1990) based on a years-long collaboration with SmallSoft through a university-industry network (Mathiassen, 2002). This approach allowed us to analyze in detail how adoption of SPI-technology evolved over time and to link the findings to the theoretical debate over control and drift (Ciborra *et al.*, 2000; Ciborra, 2002).

As suggested in Pettigrew's theory of longitudinal field research on change, the study presents a peek into an ongoing process of change. We base the study on several years of personal interaction with SmallSoft. We complement these insights with access to extensive documentation and interviews, and the study ends in a situation where management has just been through a period of breakdowns, reflections and learning, and has decided to engage in yet another SPI-initiative. This view of ongoing change leads to open-ended interpretations and conclusions. We handle the time aspect of the study by focusing on encounters punctuating relatively stable episodes of evolution (Cho *et al.*, 2006; Peterson, 1998; Newman and Robey, 1992). We have applied the criteria of contextual, processual, historical, and pluralist data to triangulate between personal experiences through involvement, documentary and archive data, observational and ethnographic material, and in-depth interviews. The authors have gained knowledge from direct involvement in the improvement efforts in SmallSoft, at different times and in different roles. These differences help to balance the presentation of the case.

SmallSoft

SmallSoft is a small Danish IT-firm with approximately 50 employees. Their core competence is to combine domain specific engineering knowledge with IT-competencies to serve their customers by developing new solutions or by adapting standard software.

SmallSoft was founded when a large engineering firm closed down in 1987 because of bankruptcy. Three engineers formed a consultancy firm operating in the same industry segment. They soon began to develop dedicated software tools in-house to support consultation, which led them to develop a material-management-tool as a joint venture with a main customer. Even though the virtues of this software product were

its built-in domain knowledge, not its technical quality, it evolved into an important standard software product and the development process laid the basis for the software practices of SmallSoft today.

In the early 1990s, software production was established as a separate business area and SmallSoft started to employ additional developers and established a new department for tailored IT-systems. After a couple of years, the IT-business area dominated the firm that had grown to include 42 developers; 25 developers worked in the Standard Department, responsible for the original and still important standard software that over the years had developed into a portfolio of subsystems and versions; 15 developers worked in the Tailored Department, tailoring systems to a variety of customers; finally, a new two-person department developed a storage management product based on a recent acquisition.

Since the three founding engineers had just watched a bankruptcy from the inside, they developed a firm culture based on a low risk attitude and a belief in core engineering skills and long-term customer relations. During the difficult first years, standards for managing, staffing, and building customer relations and solutions were established. Despite growth, SmallSoft's top level management and board continued to exercise a defensive business and investment strategy, expecting surplus every single month. Customer relations were built on long-term personal relations and employees were valued as experts, since most development work demanded detailed domain knowledge. Software engineering skills were generally considered less important. During the mid-nineties, the development of the standard software product was fleshed out from development of customer versions to overcome severe quality problems within the Standard Department. This subgroup of developers, dedicated to the standard software, found software skills, software processes, and product quality increasingly important.

Data collection

Data were collected covering a period of ten years, where we periodically collaborated with SmallSoft as employees, as university teachers, and as researchers engaged in action research (Mathiassen, 2002; McKay and Marshall, 2001; Susman and Evered, 1978). Our relationship with the firm began in 1999 when one author was employed as project leader, participating in the implementation of the new quality assurance system (QA-system) (launched in 1996) and ended in 2005 when an action research project was concluded. Through these activities, it was possible to collect a diverse and rich selection of data from a variety of sources covering a relatively long period of time. First, as part of a university-based action learning program (see section SPI-Action Learning) a group of employees at SmallSoft engaged in an SPI-initiative, starting in 2000 and supervised by one of the authors; the resulting report included assessments of maturity, analysis of practice, descriptions of interventions, and a new SPI-organization. Second, in an interview in March 2004 the key manager at SmallSoft described and reflected on important events in the firm's quality assurance (QA) initiative from 1996 to 2004. Third, in fall 2005 we interviewed other key actors about their view on SPI-technology adoption. The fourth main source was the other author's participation in an action research project with SmallSoft from fall 2003 to fall 2005; this initiative provided e-mail correspondence, documents, notes and minutes from meetings, plus extensive research notes. Adding flesh to these primary data

sources, we also had access to the firm's internal SPI-document archive, graduate student reports from collaboration with the firm, recordings from meetings and work situations inside the firm, and interviews with research colleagues. Finally, we had extensive experiences from our personal participation in the process of adopting SPI-technology at SmallSoft.

Two aspects of this data collection process require special attention. First, part of the data came from the authors' engagement in an action learning project and an action research project; this direct participation introduces possible bias in selecting and interpreting data. Second, part of the data draws on interviews with different stakeholders in which the interviewees were asked to retrospectively reflect on important SPI-technology adoption. While the use of retrospective data in process studies is quite common, it involves the risk that interviewees forget or rationalize what originally happened. To reduce the adverse effect of these factors we took advantage of our access to many different complementary sources to triangulate findings (Yin, 1994).

Data analysis

Initially, we combined Newman and Robey's encounter-episode distinction (1992) with Ciborra's control-drift distinction (Ciborra *et al.*, 2000; Ciborra, 2002) to form a historical map covering a time span of ten years (1996-2005). Working systematically through the data sources, we identified candidate encounters by selecting events that were either mentioned in the interviews as important, or that we found to have had significantly impacted the adoption process. In doing so, we focused on events that were caused by management's attempts to control SPI-technology adoption and on events that occurred in SmallSoft's environment. As an initial step towards a process analysis, we mapped the identified encounters according to the chronological timeline and described them briefly together with the intermediary episodes.

Subsequently, we went deeper into the data and started to analyze each encounter-episode. In doing so, we focused on SPI related activities, on their impacts on software development practices, and on activities and events in SmallSoft's environment relevant to the adoption of SPI-technology. As these more detailed analyses progressed, we iterated the selection and description of encounters and episodes until a satisfactory analysis emerged. Hence, as we analyzed how SPI-technology adoption at SmallSoft was shaped by actors with different interests over time, we constantly looked for evidence that our selection of encounters reflected significant events in shaping SPI-technology adoption at SmallSoft. These efforts led to confirmation of some encounters; but we also had to discard, modify, or replace other encounters.

Finally, to make sense of how SPI-technology was adopted at SmallSoft viewed from the perspective of control and drift, we identified two intrinsically related socio-technical networks. First, there was the relatively stable and powerful production-network in which managers and software developers across SmallSoft's three departments developed new solutions in response to customer requests. Second, there was the less stable and weaker improvement-network through which a small group of different actors over time attempted to improve practices in the production-network through the adoption of new development technologies. These two networks offer complementary perspectives on Smallsoft's adoption of

SPI-technology, one focusing on production of software and the other on the ongoing SPI-efforts. In line with Actor Network Theory (Latour, 1987; Callon, 1986; Callon and Law, 1989; Law, 1991; Walsham, 1997), each network includes actors within the firm, ways of working, current and emerging technologies, as well as relationships to forces outside the firm.

Working through the encounters and data once again we wrote up a coherent account of SPI-technology adoption and of how it was shaped through managerial control and improvisations triggered by unpredictable events in the environment. While the choice of encounters at this point had stabilized, our interpretations changed as we negotiated the final version of the story between the data, the theoretical framing, and the experiences and knowledge of the two authors.

Process analysis

The adoption of SPI-technology in Smallsoft passed through seven encounters and subsequent episodes during 1996-2005. This chronology is summarized in Table I.

ISO-9001 Certification

The QA-effort was initiated in 1996 and successfully completed in 1998 with an ISO-9001 certificate. Smallsoft's QA-system was developed by a group consisting of a department manager operating as QA-coordinator and two key developers representing both management interests and engineering practices. The QA-coordinator's interest was to adopt best practices firm-wide by including these in the QA-system as mandatory procedures. Top-level management focused, however, on the certificate; they believed it would help to promote Smallsoft as a professional software house. The QA-system was implemented as an on-line library of procedures, checklists, and templates supplemented with an internally developed requirement, configuration, and change management tool. No permanent QA-group and no standard for operating the QA-system or other quality techniques were implemented. Project managers were expected to comply with the QA-system and all employees received introductory training in an attempt to spark the improvement of practice in Smallsoft's production-network.

After the certification, management left no doubt about the order of priorities in Smallsoft as successful sales resulted in increased workloads; everybody's focus was on producing software and serving customers. There was no time and energy left to institutionalize procedures or reflect on resulting practices. Neither the QA-group, nor management followed up on the implementation of the QA-system. The newly formed improvement-network was largely reduced to the QA-system itself and did not bring any noteworthy changes into practice. However, in the Standard Department, severe problems with configuration management were solved by implementation of a new tool into the QA-system; the tool was initially developed by employees in response to personal needs. When the tool was introduced firm-wide it met no resistance. As a result, configuration management practices changed quickly and led to better customer relations.

There was no doubt that the certification process improved the quality of the software process in Smallsoft, but management and the employees very quickly lost interest in quality assurance after the successful certification.

	SPI-activity	SPI-organization	Immediate impact	Subsequent impact
ISO 9001 Certification (1996-1998)	Designing ISO-9001 QA system	An *ad hoc* representative QA-group (two employees and the QA-coordinator)	Successful ISO-9000 certification and implementation of new configuration management tool	Integration of QA-system into practice fails. Sustained use of configuration management tool
SPI-Action Learning (Spring 2000 to January 2001)	Participating in action learning program, in project planning, and SPI	An *ad hoc* action learning group (three key employees). QA coordinator internal sponsor	Successful learning as new knowledge and energy was fed into the firm. No change of practices	Assessments planned and possible improvements identified
SPI-Pilot Projects (Autumn 2000 to March 2001)	Assessing processes, planning and designing new process, conducting SPI pilots	An *ad hoc* action learning group (three key employees). Two production projects as pilots. Top management and supervisor in steering committee	Successful CMM assessment. Knowledge of appropriate change practice gained. New processes designed and tested	New knowledge of SPI and organizational change acquired. Sporadic quality review practices
Forming SPI Organization (January 2001 to June 2001)	Designing, negotiating, and deciding on new centralized SPI organization	A new centralized SPI group appointed (two from action learning group and QA coordinator). *Ad hoc* PIT groups for each SPI initiative	Successful design and decision on a new SPI –organization. Members appointed to SPI-group that started working	No change of practice. Successful evaluation of organizational change documented
SPI-Champion Exit (November 2002 to Spring 2004)	Participating in SPI-action research project. Forced to focus on sales activities ended the SPI-activities	Up until breakdown: The same central SPI-group with action researchers as consultants. After breakdown: No SPI-organization	The SPI-champion left the firm, partly because of the breakdown of SPI-initiatives	All SPI-initiatives stop
Learning from SPI-Failure (Summer 2004 to Summer 2005)	Management realizing SPI-breakdown and starting to reflect on present and past improvement practices	No SPI-organization. QA-coordinator still engaged personally	QA-coordinator develops new understanding of SPI-practice. Renewed collaboration with researchers	Ongoing reflections on ideas and research to improve SPI efforts
A Grassroots Approach (November 2005)	Designing and implementing an SPI PIT-organization	A permanent PIT organization implemented. All employees participate in improvement team. QA coordinator is sponsor	Successful and enthusiastic design and implementation of new grassroots SPI organization	Three PITs (process improvement team) working. Eighteen employees and managers involved in improvement network. Future success unpredictable

Table I.
Chronology of SPI-technology adoption

SPI-action learning

In spring 2000, the local university offered Smallsoft an action learning education (Mathiassen *et al.*, 1999) in project management and improvement. Smallsoft's management welcomed this opportunity for inexpensive improvements and granted participation of three key employees (the action learning group). The education would engage them in an informal university-industry-network and management hoped that this would develop new relationships within the region, boost Smallsoft's image as a professional firm, and provide a useful update on software engineering. Management left the initiative to the action-learning group and kept its own focus on the production-network.

Theory was taught at the university, and action learning was conducted as supervised projects, applying theory to support interventions in each participating firm. While attending the course, the action-learning group studied project management and SPI theory and engaged in discussions with teachers and fellow students from other firms.

The action-learning group identified difficulty prioritizing improvement work in relation to everyday production work as the main threat to successful SPI in Smallsoft. Supported by their academic supervisor, the group decided to formally establish an improvement-network as a counterweight to the production-network and as a means to more effectively involve management and colleagues in SPI-activities.

SPI-pilot projects

The action-learning group agreed with management to conduct interventions according to the IDEAL-model (McFeeley, 1996). In the diagnosis phase, they assessed Smallsoft's process maturity through a simplified CMM-like questionnaire focused on CMM level two: requirement management, project planning, project tracking and oversight, subcontract management, quality assurance, and configuration management.

Smallsoft failed the assessment, fulfilling less than half of the requirements for the areas "project tracking and oversight" and "quality assurance", and up to 70 per cent for "configuration management". The scores surprised the action learning group, since requirement management and project planning were emphasized in the QA-system offering procedures, standards, and IT-based tools for both.

The action-learning group presented the assessment to management and their supervisor. Management wanted to keep investments low by integrating previously planned improvements on project planning, and the group agreed on two interventions: "Project planning, tracking, and oversight" and "quality assurance". At the same meeting, the concept of "quality meetings", implementing external reviews throughout a project's life-cycle, was created and agreed on as the QA-initiative.

The action-learning group wanted to facilitate SPI by testing the new processes, by gaining improvement experience, and by winning ambassadors for the improvement-network. Hence, interventions into two pilot projects were planned by the action learning group following the management policy of no extra workload for pilots. The group planned four meetings with each pilot, two of which prepared and planned the intervention while two practiced the new processes under the group's supervision.

The introductory discussion of assessment results and participation in planning the interventions created a positive attitude. However, applying the newly designed

procedures and templates for project planning was a failure. The first pilot had already committed externally to a plan, so they preferred instead to focus on risk and stakeholder analysis. The other pilot planned according to the proposed procedure, but shortly after, the customer wanted to turn the project into a flexible prototyping exercise. Both initiatives failed, simply because they lacked coordination with the stronger production-network.

The introduction of quality meetings was more successful in both pilots. The action learning group used a workshop to introduce a coherent, flexible, and simple one-page tool to support planning, conducting, and documenting quality meetings. Most developers were interested in more management attention on individual projects and supported the new practice, confident that it would improve products and practice. The first quality meeting in both pilots led to improvements in plans and products, and the participants found the practice simple, useful, and effective.

The pilots were evaluated both by management and as part of the education and the results contributed to the forming of a new SPI-organization for Smallsoft.

Forming the SPI-organization

While engaged in the pilots, the action-learning group negotiated a plan for continued SPI with management over three meetings. As a result, management gradually adopted SPI as a way to promote Smallsoft as a professional software house thus supporting the improvement-network.

After the diagnosis in October 2000, management was introduced to SPI by the university supervisor and committed to participate in a workshop on further adoption of SPI-technology in Smallsoft. At the workshop, the action-learning group provided management with a status of the pilots, an overview of all existing improvement initiatives at Smallsoft, and a tailored proposal for SPI in Smallsoft based on CMM. The group had recorded 25 improvement initiatives that were either planned or so-called "bubblers" – improvements spontaneously initiated within a department and somewhat "bubbling up from below". The "bubblers" were a well-known part of the Smallsoft culture, as employees were expected to do their best and change practices if needed. This high degree of delegation led over time to important improvements of software practices, starting as local or even individual initiatives.

Management approved the proposal as interesting and feasible, but asked for more detailed planning before they could commit themselves. In the beginning of March 2001, when the pilots were completed, the action-learning group presented a detailed description of an SPI-organization for Smallsoft. The group suggested limiting innovation to project management, but top-level management wanted to include software engineering and customer relations and increased the man hours for this work from 500 to 800 per year. Revising and implementing practices tested in the pilots and preparing for a new CMM assessment were the first tasks. The SPI organization (Figure 1) was centered around an SPI-group, coordinating and planning initiatives handled by dedicated process improvement teams (PITs) consisting of members of the SPI-group and interested developers.

It was planned to present the new SPI-organization and the SPI-group at the next employee meeting in April 2001. Due to the well-prepared proposal, the comfortable order situation, and the positive experience with the QA-system, management was very supportive and proclaimed that Smallsoft would reach CMM level three in three

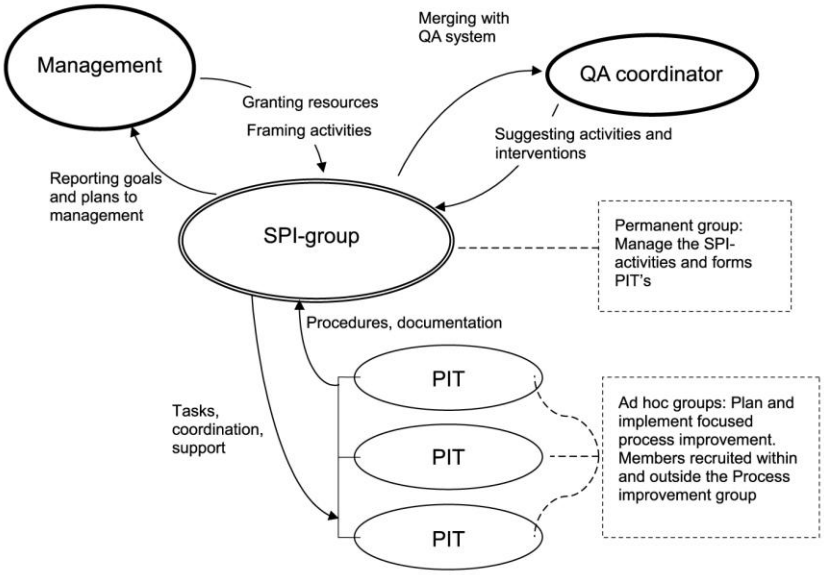

Figure 1.
SPI-organization based on CMM

years. However, the QA-coordinator and the SPI-group were skeptical and the formation of the SPI-group made some experienced developers feel undercut in their professional authority.

The SPI-group included the QA-coordinator and employees with extensive SPI-knowledge. All members were, however, deeply involved in the production-network and could hardly find time or energy for SPI. When the group met for the first time in June 2001, members were already running late according to plans and another meeting was required in August before they – six months late – presented the SPI-organization at a staff meeting in November.

One member acted as SPI-champion; he deeply believed in the quality meeting concept as a realistic way to improve practices and products. He pushed the concept as part of the August meeting agenda and presented detailed and revised quality meeting practices. His enthusiasm persuaded the other members to implement the concept in their departments before the next SPI-group meeting. He personally conducted the first quality meeting in September 2001.

In summary, while management believed SPI-technology adoption was on track, this period was characterized by low SPI-activity and neglect or even hostility from developers outside the improvement-network. The SPI-group could hardly find time to meet and the SPI-champion did not succeed in engaging the other members. No additional developers were involved since no PITs were formed.

Priorities in the firm still favored the production-network, but the QA-coordinator, having no energy left for SPI-initiatives, still looked for ways to empower the improvement-network. In fall 2002, he engaged a student group from the local university to analyze the ongoing SPI efforts. The students recommended decentralization of SPI, based on self-improving software teams, due to the customer

centered culture and the high degree of delegation at Smallsoft. The analysis was discussed with the QA-coordinator, but it had no visible influence on the improvement-network. In fact, adoption of SPI-technology had failed and the improvement-network continued to be characterized by rather weak and heterogeneous interests.

SPI-champion exit

Early 2003, Smallsoft entered an action research project with the local university hoping to revitalize SPI-technology adoption. The researchers and the SPI-group planned SPI-activities such as a best practice survey, implementation of code review procedures, and improvement of the quality meeting practice (five quality meetings were conducted during the first six months of 2003). The amount of activities was overwhelming for the SPI-group, but since the SPI-champion hoped the collaboration could bring about real change, he dedicated personal energy and working hours for others.

During fall 2003, Smallsoft experienced a decline in market and intensified their sales efforts. They had to downsize for the first time ever in spring 2004. The situation drained energy and took the focus away from the renewed SPI-effort, and Smallsoft postponed or cancelled most activities. The SPI-champion became increasingly frustrated and finally left the firm. The champion disagreed with management priorities and believed that a strong improvement-network was crucial for the long-term survival of the firm. As a result, all SPI-activities stopped, but the research project manager pushed Smallsoft to either re-engage in or to leave the research project. This external pressure led to negotiations of continued collaboration.

Learning from SPI-Failure

The failure of the centralized, norm-based SPI-initiative made the QA-coordinator reflect on the situation. At this point, in fall 2004, he defined the central organization as a complete failure:

> We wanted to have a central SPI-group that would coordinate and manage SPI-activities. We told people to forward their ideas for improvement to us, and then we would try to implement them in practice. It was a complete failure (interview with the QA-coordinator)

He pointed to three reasons for the failure: lack of time in the SPI-group, neglect from everybody else, and the need to maintain monthly economic surplus for the firm limiting investments in innovation. The QA-coordinator was, however, still committed to the improvement-network, since he believed that continuous improvements were necessary to sustain the firm.

He realized the dominance of the production-network over the improvement-network and reflected on how successful improvements had been driven by production needs, thus creating overlap between the production-network and the improvement-network. Several improvements had been implemented during the relatively short lifetime of the firm, especially prior to the formation of the SPI-group. These improvements were often initiated by developers experimenting with innovations and subsequently spread by word of mouth ("bubblers"). Two of the most significant; the configuration management system and the QA-system, were initiated this way before they were upgraded to firm level and granted resources. After the SPI-group was formed, employees and management still reflected on and learned from the SPI-experiences, but hardly focused on adapting personal practice accordingly.

However, improvements did appear i.e. a better quality of project planning and an increased frequency of code-reviews.

> I started worrying if forming a central SPI-group had killed the grassroots improvements, and if no activity in the SPI-group, would result in no improvements at all (interview with the QA-coordinator)

The QA-coordinator started to advocate an approach to SPI where improvements were to be initiated from the grassroots of the firm (individually or locally), driven by personal involvement of employees. This idea was in line with the earlier recommendations from the student group. Three considerations led him to the following conclusion: small firms cannot afford to invest in a dedicated improvement unit; the centralized SPI-initiative had implied increasing overhead; and a centralized SPI-group would not know established practices well enough to propose realistic and efficient changes. The researchers agreed to investigate how a grassroots approach to SPI could be facilitated and implemented so that the improvement-network was more in line with the production-network at Smallsoft.

A grassroots approach

Projects and individuals of Smallsoft had proven capable of improving practices, but the exploration of grassroots approaches to SPI raised the question of how local improvements spread firm wide. In December 2004, the researchers conducted a social network analysis of the improvement-network, displaying very close connections within each department, but only loose connections across department borders and to top level management (Nielsen and Tjørnehøj, 2005). Thus, Smallsoft had a highly developed network for local improvements, but no basis for sharing knowledge and practices across departments. The researchers suggested either making SPI-initiatives local to departments losing the benefits from sharing knowledge, or building relations between departments to support knowledge sharing.

These insights strongly impacted an SPI-strategy meeting in March 2005, engaging two department managers (including the QA-coordinator) and top level management in a sincere discussion of experiences with SPI, the QA-system, and the problems and benefits of having self-governed developers and departments. In the light of a sales boom, making it difficult for the new department in order to keep up with demands, management felt they could learn from the two established departments to avoid well-known failures. Thus, knowledge sharing between departments was given priority within Smallsoft.

One of the researchers had just learned how a reputed Danish firm had successfully moved to level 3 in CMM by involving all employees in PITs. This success inspired Smallsoft to form eight improvement initiatives across departments in a matrix-like PIT organization. Management decided to engage all developers in one or more PITs and, as a strong sign of management commitment, they assigned man-hours to the task. Employees were assigned during summer 2005 to ensure participation from all departments and more projects in each PIT. In August 2005, the PIT organization was launched at a kickoff meeting. Most developers participated and engaged in discussing the implications of the new approach. The urgent PITs started immediately while less urgent ones would start within a year. The PITs were self-organized and autonomous with respect to improvement directions.

In November 2005, three out of eight PITs were active, but working at a slower pace than expected. The PITs found major differences in practices between the three departments. Members from the established departments were expected to transfer practices to the new department, but the new department had its own traditions and preferences. Despite these challenges, the PITs made progress and arrived at realistic solutions. A total of 18 developers and parts of management were now participating in an improvement-network that seemed to grow stronger than ever before during adoption of SPI-technology at Smallsoft.

Discussion

Smallsoft did indeed experience the difficulties that the literature identifies as being related to adoption of SPI-technology; lack of resources and SPI-knowledge (Steel, 2004; Brouse and Buys, 1999; Villalon *et al.*, 2002; Brodman and Johnson, 1994; Kilpi, 1998; Saastamoinen and Tukiainen, 2004); sensitivity towards changing environments (Ward *et al.*, 2001); and incongruity between small firm culture and SPI-theory (Leung and Yuen, 2001; Varkoi *et al.*, 1999). However, while the literature primarily suggests different forms of downsizing and pragmatic usage of CMM or alternative SPI-technologies designed for small firms, Smallsoft handled these challenges differently.

SPI-adoption at Smallsoft

Smallsoft dealt with the lack of resources and knowledge through networking with the university and local firms (Kautz, 1998). They took advantage of opportunities in their environment and invested in low-cost attempts to adopt SPI-technology without jeopardizing their need to produce surplus every month. However, outcomes always came down to the daily struggle between the everyday tasks of developers and the extra workload and risk that changing habits and work practices implied, even when resources were granted from management (Mustonen-Ollila and Lyytinen, 2003).

Table I summarizes the presented encounters with SPI-technology adoption at Smallsoft followed by episodes focused on everyday production work. In the short term, each SPI-initiative was successful; developers and managers found them useful and promising; but in the subsequent episode, SPI-innovations were overshadowed by production concerns so effects seldom materialized as planned.

However, employees and management did reflect on the experiences, internalized some of the theories, methods and techniques, and improvements did emerge over time, but not as planned results of firm level decisions, rather as local or individual changes in practice. This was underpinned in the interviews with the key actors in fall 2005, when we investigated if any improvements had taken place. They emphasized both personal learning and more concrete examples of improvements emerging from the encounters, e.g. matured project planning and tracking and reviews to increase quality of code. Also the "Learning from SPI-failure" encounter was emphasized by the QA-coordinator as a major breakthrough for Smallsoft's SPI efforts.

The ongoing struggle to develop and sustain an improvement-network along with the dominating production-network involved managers and employees; it crossed organizational borders, it was reinforced through different forms of technology, and it was heavily influenced by external events and forces. Throughout, actors tried to persuade colleagues to see SPI as a long-term contribution to their production interests, and management (in particular the QA-coordinator) continued to support the

improvement-network through controlled initiatives, insisting that SPI was important for Smallsoft. These efforts were, as described above, mainly successful when improvements were perceived by employees as concrete and sustainable extensions of the production-network.

The "Grassroots approach" following the "Learning from SPI-failure" encounter was indeed based on the experiences from Smallsoft's ongoing struggle to adopt SPI-technology. It exploited individual and local initiatives in a new organization that supported improvements through management decisions on subjects, priorities, and resources and through negotiation of new practices between the everyday users representing the production network. Measured on the traditional short term scale, this initiative was successful, but it remains yet to be seen whether it will bring sustainable changes.

Between control and drift

According to Ciborra (2002), control is based on a rational view of the world in which managers understand and plan events by applying simplistic theoretical models to decisions and practices. Drift, on the other hand, emphasizes how side effects, bricolage, hacking, formative context, and people's everyday coping make reality drift away from plans, thereby opening up to options and innovations that were otherwise unthinkable.

Smallsoft's adoption of SPI-technology displays both control and drift elements (Ciborra, 2002; Ciborra *et al.*, 2000). However, contrary to Ciborra, the two elements interacted, their relative dominance shifted as the process unfolded, and both elements had positive impacts on the adoption of SPI-technology. Rather than seeing control and drift as alternative management philosophies, the Smallsoft experience suggests that these are complementary, and intrinsically related opposites of a dialectical relationship (Bjerknes, 1991; Van de Ven and Poole, 1995; Mathiassen, 1998; Robey and Boudreau, 1999). While SPI-adoption at Smallsoft was shaped by several, interacting contradictions (e.g. between developers and managers, between different departments and customers, and between espoused theories and theories-in-use (Argyris and Schön, 1978)), we found the contradiction between control and drift to be particularly helpful in making sense of the reported ten years of SPI-technology adoption.

Initially, the principal aspect of the contradiction was control when Smallsoft chose a centralized and controlling improvement approach to their ISO-9001-certification, while drifting had less influence on the result through the personal hacking of a configuration management system. In the following episode, however, management believed to have laid out the direction and did not exercise control through systematic follow up on the QA system. As a result, drifting dominated and developers did not comply with the procedures unless they were immediately useful. Only when the certificate was renewed did the ISO-9001 assessments reinforce control.

When Smallsoft unexpectedly was offered SPI-action learning, a change initiated by drift, management engaged to regain control over and create momentum for the SPI-process. The formal organization of the education and the rational worldview of the theories taught and later applied emphasized control both during this encounter and the SPI-pilot projects. While some drifting appeared as a result of pressures from customers, control dominated based on influences from management, the

action-learning group, the teachers, the adopted theories, and the experiences from other influential Danish firms.

The SPI-organization was also initially conceived as a centralized decision and control mechanism, involving experts, management, and internal change agents, and leaving only limited room for everyday coping, unexpected events, hacking, or unplanned innovations. The proposed organization was at first broadly accepted, but then by and large ignored. In the following episode the power of controlling forces diminished because the SPI-group struggled to find time for improvements due to everyday working responsibilities in their departments. As a result, the SPI-organization and – activities drifted away from what was intended.

The attempt to control SPI-adoption by entering the action-research project was effectively stopped by drifting forces from outside Smallsoft. As the market declined and the economic situation at Smallsoft got worse, all planned improvement initiatives were put on hold and the SPI-champion left the firm. Following this breakdown, management realized that one-sided rational attempts to control SPI were costly for Smallsoft, because the benefits of drifting forces, i.e. effective bricolage, hacking, and tinkering from self-organized "bubblers" and unplanned events, almost disappeared. On the other hand, sharing knowledge, best practices, and strategic goals across departments was important for Smallsoft to stay competitive and introduced needs for exercising some control.

The "unthinkable" innovation, the Grassroots Approach, gave management a sense of control while leaving amble space for negotiating new practices based on everyday coping and survival strategies developed by employees who volunteered to engage in PITs. Though management still exercised control by prioritizing focus areas, by granting resources, by appointing experts, and by requesting monthly reporting, they also embraced drifting to help develop and share successful innovations.

This account of the struggle between control and drift illustrates how the two opposites interacted and both positively impacted the ongoing SPI-adoption at Smallsoft. New SPI-theories and -models offered control approaches and brought new knowledge to management, internal SPI-agents, and employees. Management's attempts to control SPI-adoption framed collaborative experimenting and learning that kept the improvement-network alive. Management's continued control efforts kept Smallsoft vigilant by insisting on and pushing the organization towards change.

At the same time, the options unexpectedly offered from outside the firm, such as education, student involvement, and the research project, provided Smallsoft with resources and knowledge to effectively engage in improvements. The employees' everyday coping, bricolage and hacking helped adapt SPI-technology to the everyday reality of the firm by stopping failing initiatives, by aligning improvement initiatives with the production-network, and by creating unplanned innovations. Hence, while continued efforts to control created momentum, informed the process, and provided strategic direction, drifting forces secured real world results by making changes take place in the production-network.

Hence, the Smallsoft experience suggests that managers of small software firms remain flexible and constantly negotiate technology adoption practices between control and drift; creating momentum and direction according to firm goals through attempts to control, while at the same time exploring backtalk, options, and innovations from drifting forces inside and outside the firm. In such a dialectical

approach to technology adoption, control and drift represent complementary and intrinsically related approaches that, if appropriately negotiated, support each other and lead to acceptable adoption outcomes.

Note

1. Ciborra explains bricolage as the activity when humans leverage the world as defined by the situation, improvisation as dealing with sudden unpredictable interventions and hacking as an ingenious activity that through iterations, reuse, and reinterpretations of the existing programming environment leads to the implementation of new solutions. Ciborra (2002).

References

Aaen, I., Arendt, J., Mathiassen, L. and Ngwenyama, O. (2001), "A conceptual map of software process improvement", *Scandinavian Journal of Information Systems*, Vol. 13, pp. 81-101.

Ares, J., García, R., Juristo, N., López, M. and Moreno, A.M. (2000), "A more rigorous and comprehensive approach to software process assessment", *Software Process: Improvement and Practice*, Vol. 5, pp. 3-30.

Argyris, C. and Schön, D.A. (1978), *Organizational Learning: A Theory of Action Perspective*, Addison-Wesley Publishing Company, Reading, MA.

Balla, K., Bemelmans, T., Kusters, R. and Trienekens, J. (2001), "Quality through managed improvement and measurement (QMIM): towards a phased development and implementation of a quality management system for a software company", *Software Quality Journal*, Vol. 9, pp. 177-93.

Batista, J. and Figueiredo, A.D.D. (2000), "SPI in a very small team: a case with CMM", *Software Process: Improvement and Practice*, Vol. 5, pp. 243-50.

Bennetts, P.D.C. and Wood-Harper, A.T. (1998), "The soft system methodology as a framework for software process improvement", *Journal of End User Computing*, Vol. 10, pp. 12-19.

Bjerknes, G. (1991), "Dialectical reflection in information systems development", *Scandinavian Journal of Information Systems*, Vol. 3, pp. 55-77.

Brigham, M. and Introna, L.D. (2006), "Hospitality, improvisation and Gestell: a phenomenology of mobile information", *Journal of Information Technology*, Vol. 21, pp. 140-53.

Brodman, J.G. and Johnson, D.L. (1994), "What small businesses and small organizations say about the CMM", paper presented at the 16th International Conference on Software Engineering, Sorrento.

Brouse, P.S. and Buys, R.T. (1999), "Affordable ways to improve application development", *IT Professional*, Vol. 1, pp. 47-52.

Börjesson, A. and Mathiassen, L. (2003), "Making SPI happen: the IDEAL distribution of effort", paper presented at the 36th Annual Hawaii International Conference on System Sciences, IEEE Computer Society, Big Island, Hawaii.

Börjesson, A. and Mathiassen, L. (2005), "Improving software organizations: the agility challenge", *Information, Technology & People*, Vol. 18, pp. 359-82.

Callon, M. (1986), "Some elements of a sociology of translation: domestication of scallops and the fishermen of St Brieuc Bay", in Law, J. (Ed.), *Power, Action and Belief: A New Sociology of Knowledge?*, Routledge & Kegan Paul, London.

Callon, M. and Law, J. (1989), "On the construction of socio technical networks: content and context revisited", *Knowledge and Society*, Vol. 9, pp. 57-83.

Casey, V. and Richardson, I. (2004), "A practical application of the IDEAL model", *Software Process: Improvement and Practice*, Vol. 9, pp. 123-32.

Cho, S., Mathiassen, L. and Nilsson, A. (2006), *Event-Based Actor Network Analysis of IT-Based Change*, Georgia State University, Atlanta, GA.

Ciborra, C.U. (1994a), "From thinking to tinkering", in Ciborra, C.U. and Jellasi, T. (Eds), *Strategic Information Systems*, John Wiley and Sons, Chichester.

Ciborra, C.U. (1994b), "A platform for surprises: the organization of global technology strategy at Olivetti", *IFIP WG8.2 Working Conference on Information Technology and New Emergent Forms of Organizations: Transforming Organizations with Information Technology*, North-Holland Publishing Co., Amsterdan.

Ciborra, C. (1999), "Notes on improvisations and time in organizations", *Accounting, Management and Information Technologies*, Vol. 9, pp. 77-94.

Ciborra, C. (2002), *The Labyrinths of Information: Challenging the Wisdom of Systems*, Oxford University Press, New York, NY.

Ciborra, C. and Hanseth, O. (1998), "From tool to Gestell – agendas for managing the information infrastructure", *Information Technology & People*, Vol. 305 -327.

Ciborra, C. and Lanzara, G.F. (1994), "Formative contexts and information technology: understanding the dynamics of innovation in organizations", *Accounting, Management and Information Technologies*, Vol. 4, pp. 61-8.

Ciborra, C.U., Braa, K., Cordella, A., Dahlbom, B., Failla, A. and Hanseth, O. (2000), *From Control to Drift: The Dynamics of Corporate Information Infrastructures*, Oxford University Press, Oxford.

Coleman, G. and Verbruggen, R. (1998), "A quality software process for rapid application development", *Software Quality Journal*, Vol. 7, pp. 107-22.

Dion, R. (1992), "Elements of a process-improvement program", *IEEE Software*, Vol. 9, pp. 83-5.

Dybå, T. (2003), "Factors of software process improvement success in small and large organizations: an empirical study in the Scandinavian context", *ACM SIGSOFT Software Engineering Notes*, Vol. 28, pp. 148-57.

El-Emam, K., Goldenson, D., McCurley, J. and Herbsleb, J. (2001), "Modeling the likelihood of software process improvement: an exploratory study", *Empirical Software Engineering*, Vol. 6, pp. 207-29.

Elbanna, A.R. (2006), "The validity of the improvisation argument in the implementation of rigid technology: the case of ERP systems", *Journal of Information Technology*, Vol. 21, pp. 165-75.

Grechenig, T. and Zuser, W. (2004), "Creating organic software maturity attitudes (COSMA) selected principles and activities for software maturity in small and medium software enterprises", paper presented at the 4th International Conference on Quality Software (QSIC'04). Braunschweig.

Hansen, B., Rose, J. and Tjørnehøj, G. (2004), "Prescription, description, reflection: the shape of the software process improvement field", in Powell, P. and Grant, K. (Eds), *UK Association of Information Systems Conference*, Glasgow Caledonian University, Glasgow.

Hanseth, O., Ciborra, C. and Braa, K. (2001), "The control devolution: ERP and the side effects of globalization", *SIGMIS Database*, Vol. 32, pp. 34-46.

Harjumaa, L., Tervonen, I. and Vuorio, P. (2004), "Using software inspection as a catalyst for SPI in a small company", *Product Focused Software Process Improvement*, Springer, Berlin/Heidelberg.

Horvat, R.V., Rozman, I. and Györkös, J. (2000), "Managing the complexity of SPI in small companies", *Software Process: Improvement and Practice*, Vol. 5, pp. 45-54.

Humphrey, W.S., Snyder, T.R. and Willis, R.R. (1991), "Software process improvement at Hughes Aircraft", *IEEE Software*, Vol. 8, pp. 11-23.

Jakobsen, A.B. (1998), "Bottom-up process improvement tricks", *IEEE Software*, Vol. 15, pp. 64-8.

Kautz, K. (1998), "Software process improvement in very small enterprises: does it pay off?", *Software Process: Improvement and Practice*, Vol. 4, pp. 209-26.

Kautz, K.H. (1999), "Making sense of measurement for small organizations", *IEEE Software*, Vol. 16, pp. 14-20.

Kautz, K.H., Hansen, H.W. and Thaysen, K. (2000), "Applying and adjusting a software process improvement model in practice: the use of the IDEAL model in a small software enterprise", *22nd International Conference on Software Engineering*, ACM Press, Limerick.

Kehoe, R. and Shah-Jarvis, A. (1995), *ISO 9000-3: A Tool for Software Product and Process Improvement*, Springer-Verlag, New York, NY.

Kelly, D.P. and Culleton, B. (1999), "Process improvement for small organizations", *Computer*, Vol. 32, pp. 41-7.

Kilpi, T. (1998), "Product management challenge to software change process: preliminary results from three SMEs experiment", *Software Process: Improvement and Practice*, Vol. 3, pp. 165-75.

Kuvaja, P., Palo, J. and Bicego, A. (1999), "TAPISTRY – a software process improvement approach tailored for small enterprises", *Software Quality Journal*, Vol. 8, pp. 149-56.

Latour, B. (1987), *Science in Action: How to Follow Scientists and Engineers through Society*, Harvard University Press, Cambridge, MA.

Law, J. (1991), "Introduction: monsters, machines, and socio technical relations", in Law, J. (Ed.), *A Sociology of Monsters: Essays of Power, Technology and Domination*, Routledge, London.

Leung, H.K.N. and Yuen, T.C.F. (2001), "A process framework for small projects", *Software Process: Improvement and Practice*, Vol. 6, pp. 67-83.

Lévi-Strauss, C. (1962), *La pensée sauvage*, Librairie Plon, Paris.

Lévi-Strauss, C. (1972), *The Savage Mind*, Oxford University Press, New York, NY.

McFeeley, B. (1996), *IDEAL: A User's Guide for Software Process Improvement*, Software Engineering Institute, Pittsburgh, PA.

McKay, J. and Marshall, P. (2001), "The dual imperatives of action research", *Information Technology & People*, Vol. 14, pp. 46-59.

Mathiassen, L. (1998), "Reflective systems development", *Scandinavian Journal of Information Systems*, Vol. 10, pp. 67-118.

Mathiassen, L. (2002), "Collaborative practice research. information", *Technology & People*, Vol. 15, pp. 321-45.

Mathiassen, L., Borum, F. and Pedersen, J.S. (1999), "Developing managerial skills in IT organizations: a case study based on action learning", *Journal of Strategic Information Systems*, Vol. 8, pp. 209-25.

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

Mustonen-Ollila, E. and Lyytinen, K. (2003), "Why organizations adopt information system process innovations: a longitudinal study using diffusion of innovation theory", *Information Systems Journal*, Vol. 13, pp. 275-97.

Newman, M. and Robey, D. (1992), "A social process model of user-analyst relationship", *MIS Quarterly*, Vol. 16, pp. 249-66.

Ngwenyama, O. and Nielsen, P.A. (2003), "Competing values in software process improvement: an assumption analysis of CMM from an organizational culture perspective", *IEEE Transactions on Engineering Management*, Vol. 50, pp. 100-12.

Nielsen, P. and Tjørnehøj, G. (2005), "Mapping social networks in software process improvement: an action research study", *Business Agility and Information Technology Diffusion*, Springer, Boston, MA.

Paulk, M.C. (1998), "Using the software CMM in small organizations", *The Joint Pacific Northwest Software Quality Conference and the Eighth International Conference on Software Quality*, pp. 350-61.

Paulk, M.C., Curtis, B., Chrissis, M.B. and Weber, C. (1993), *Capability Maturity Model for Software Version 1.1, Technical Report CMU/SEI-93-TR-24*, Software Engineering Institute, Pittsburgh, PA.

Peterson, M.F. (1998), "Embedded organizational events: units of process in organization science", *Organization Science*, Vol. 9, pp. 16-33.

Pettigrew, A.M. (1990), "Longitudinal field research on change: theory and practice", *Organization Science*, Vol. 1, pp. 267-92.

Richardson, I. (2001), "Software process matrix: a small company SPI model", *Software Process: Improvement and Practice*, Vol. 6, pp. 157-65.

Richardson, I. (2002), "SPI models: what characteristics are required for small software development companies?", *Software Quality Journal*, Vol. 10, pp. 101-14.

Robey, D. and Boudreau, M.-C. (1999), "Accounting for the contradictory organizational consequences of information technology: theoretical directions and methodological implications", *Information Systems Research*, Vol. 10, pp. 167-85.

Rodenbach, E., Van Latum, F. and Van Solingen, R. (2000), "SPI – a guarantee for success? A reality story from industry", *Product Focused Software Process Improvement*, Springer, Berlin.

Ruiz, M., Ramos, I. and Toro, M. (2002), "A dynamic integrated framework for software process improvement", *Software Quality Journal*, Vol. 10, pp. 181-94.

Saccol, A.Z. and Reinhard, N. (2006), "The hospitality metaphor as a theoretical lens for understanding the ICT adoption process", *Journal of Information Technology*, Vol. 21, pp. 154-64.

Schneider, K. (2002), *Experience Based Process Improvement. Software Quality – ECSQ 2002: 7th International Conference, Helsinki, Finland, 9-13 June 2002, Proceedings*, Springer, Berlin/Heidelberg.

Scott, L., Carvalho, L., Jeffery, R., D'ambra, J. and Becker-Komstaedt, U. (2002), "Understanding the use of an electronic process guide", *Information and Software Technology*, Vol. 44, pp. 601-16.

Scott, L., Jeffery, R., Carvalho, L., D'ambra, J. and Rutherford, P. (2001), "Practical software process improvement – the IMPACT project", paper presented at the 13th Australian Software Engineering Conference (ASWEC'01).

Steel, A.P.C. (2004), "Low-rigour, rapid software process assessments for small software development firms", paper presented at the 2004 Australian Software Engineering Conference (ASWEC'04).

Susman, G.I. and Evered, R.D. (1978), "An assessment of the scientific merits of action research", *Administrative Science Quarterly*, Vol. 23, p. 582.

Saastamoinen, I. and Tukiainen, M. (2004), Springer-Verlag, Trondheim, "Software process improvement in small and medium sized software enterprises in Eastern Finland: a state-of-the-practice study", paper presented at Software Process Improvement: 11th European Conference, EuroSPI.

Team, C.P. (2002), *CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1, Staged Representation (CMMI-SE/SW/IPPD/SS, V1, Staged)*, Technical Report SEI.

Team, C.P.D. (2000), *SCAMPISM, V1.0 Standard CMMISM Assessment Method for Process Improvement: Method Description, Version 1.0*, Carnegie Mellon University, Pittsburgh, PA.

Thowart, K. (1999), "The AMETIST process improvement experiment: towards efficient team development in small companies", *Software Process: Improvement and Practice*, Vol. 4, pp. 11-18.

Truffley, A., Grove, B. and McNair, G. (2004), "SPICE for Small Organizations", *Software Process: Improvement and Practice*, Vol. 9, pp. 23-31.

Van de Ven, A.H. and Poole, M.S. (1995), "Explaining development and change in organizations", *Academy of Management Review*, Vol. 20, pp. 510-40.

Varkoi, T., Makinen, T. and Jaakkola, H. (1999), "Process improvement priorities in small software companies", paper presented at Portland International Conference on Management of Engineering and Technology, Technology and Innovation Management, PICMET '99 Portland, OR.

Villalon, J., Agustin, G.C., Gilabert, T.S.F., Seco, A.D., Sanchez, L.G. and Cota, M.P. (2002), "Experiences in the application of software process improvement in SMES", *Software Quality Journal*, Vol. 10, pp. 261-73.

Walsham, G. (1993), *Interpreting Information Systems in Organizations*, Wiley, Chichester.

Walsham, G. (1995), "Interpretive case studies in IS research: nature and method", *European Journal of Information Systems*, Vol. 4, pp. 74-81.

Walsham, G. (1997), "Actor-network theory and IS research: current status and future prospects", in Lee, A.S., Liebenau, J. and Degross, J.I. (Eds), *Information Systems and Qualitative Research*, Chapman and Hall, London.

Ward, R.P., Fayad, M.E. and Laitinen, M. (2001), "Software process improvement in the small – a small software development company's most difficult challenge: changing processes to match changing circumstances", *Communications of the ACM*, Vol. 44, pp. 105-7.

Wilkie, F.G., McFall, D. and McCaffery, F. (2005), "An evaluation of CMMI process areas for small- to medium-sized software development organizations", *Software Process: Improvement and Practice*, Vol. 10, pp. 189-201.

Wohlwend, H. and Rosenbaum, S. (1994), "Schlumberger's Software Improvement Program", *IEEE Transactions on Software Engineering*, Vol. 20, pp. 833-9.

Yin, R. (1994), *Case Study Research: Design and Method*, Sage, Thousand Oaks, CA.

Corresponding author
Gitte Tjornehoj can be contacted at: gts@cs.aau.dk

C. Improvisation during Process-Technology Adoption: A longitudinal study of a software firm

Improvisation during Process-Technology Adoption: A Longitudinal Study of a Software Firm

A research article by Gitte Tjørnehøj and Lars Mathiassen

Corresponding Author:
Gitte Tjørnehøj
Department of Computer Science
Aalborg University
Selma Lagerlöfs Vej 300
DK-9220 Aalborg Øst
Denmark

Telephone: +45 99408910
Fax: +45 99409798
Email: gtj@cs.aau.dk

Improvisation during Process-Technology Adoption: A Longitudinal Study of a Software Firm

Abstract: Most software firms struggle to take advantage of the potential benefits of Software Process Improvement (SPI) as they adopt this technology into the complex and dynamic realities of their day-to-day operation. Such efforts are therefore typically fluctuating between management's attempt to control SPI technology adoption and events that causes the process to drift in unpredictable directions. To further understand how management's attempt to control the process is complemented by drifting, this paper investigates the role of improvisation in adoption of SPI technology in a Danish software firm, *SmallSoft,* over a ten year period (1996-2005). We found that micro-level and macro-level improvisations interacted, often in uncoordinated ways, to shape SPI technology adoption at *SmallSoft.* The improvisations enhanced employee creativity, motivation and empowerment, created momentum in the adoption process despite constrained resources, and, most importantly, helped adapt SPI technology to the everyday practices at *SmallSoft*. However, we also identified un-called for improvisations and outcomes that were uncoordinated with *SmallSoft*'s goals. Based on these findings we discuss how management in small software firms can exploit improvisations to facilitate adoption of complex technologies like SPI.

Keywords: Technology Adoption, Software Process Improvement, Improvisation

INTRODUCTION

The Capability Maturity Model (CMM) (Paulk et al., 1993) and the more recent CMM Integrated (The-CMMI-product-team, 2001, 2002) have initiated a new discipline of engineering management that plays a dominating role in software practice and research (Hansen et al., 2004). The literature offers a number of successful cases of how software firms adopted these models (Humphrey et al., 1991, Dion, 1992, Wohlwend and Rosenbaum, 1994, Carnegie Mellon, 2005) and the CMM's do indeed offer many useful recommendations for how to systematically assess and improve software operations. The CMMs are, however, rooted in the ideal of a rational, control-centered culture for software development (Ngwenyama and Nielsen, 2003), and although other software process improvement (SPI) approaches have been suggested, they do not differ from the CMMs when it comes to underlying values (Hansen et al., 2004). It is therefore not surprising most software firms struggle to take advantage of the potential benefits of SPI as they adopt this technology into the complex and dynamic realities of their day-to-day operation (Rodenbach et al., 2000, Mathiassen et al., 2002, Hansen et al., 2004).

For small software firms, it is particularly challenging to adopt SPI technology because the dominating approaches to SPI target large organizations (Leung and Yuen, 2001); there are few resources available for improvement in small software firms (Brodman and Johnson, 1994, Brouse and Buys, 1999, Villalon et al., 2002, Steel, 2004); it is by no means trivial tailoring SPI knowledge to their needs (Kilpi, 1998, Saastamoinen and Tukiainen, 2004); and, because it normally takes several complex and expensive initiatives to reach new maturity levels (Aaen et al., 2001). These challenges combined with sensitivity to highly dynamic environments (Ward et al., 2001, Mathiassen and Vainio, 2007) require small software firms to seek alternative approaches to exploit the potential benefits of SPI technology.

On this backdrop, we investigated the adoption of SPI technology over a ten year period from 1996 to 2005 in a Danish software firm, *SmallSoft*. The investigation is based on a longitudinal, interpretative case study (Pettigrew, 1990) of SPI related activities involving the firm and its relationship to the environment. In a first analysis of the data from *SmallSoft* (Tjornehoj and Mathiassen, 2008), we analyzed how encounters impacted engineering, management, and improvement practices within the firm (Newman and Robey, 1992, Peterson, 1998, Cho et al., 2008). We found that the adoption process fluctuated and was shaped between management's attempt to control SPI technology adoption and events that caused the process to drift in unpredictable directions (Ciborra et al., 2000). To further understand how management's attempt to control the process was complemented by drifting, this paper focuses on the particular role played by improvisation in helping resolve the intrinsic contradictions between *SmallSoft*'s environmental pressures towards innovation and the lack of time, knowledge, and other resources in the firm's day-to-day operation (Ciborra, 1999, Chelariu et al., 2002). Specifically, we investigate the following research question: "Why, when, and how does improvisation shape

the adoption of process technology in a small software firm?" Our analyses reveal that micro-level and macro-level improvisations interacted, often in uncoordinated ways, to shape SPI technology adoption at *SmallSoft*. However, while the improvisations had obvious benefits (e.g. enhanced employee motivation, increased momentum, and adaptation of SPI technology to the everyday practices at *SmallSoft*), we also identified un-called for improvisations and outcomes that were uncoordinated with *SmallSoft*'s goals. Based on these findings we discuss how management in small software firms can exploit improvisations to facilitate adoption of complex technologies like SPI.

The paper is structured as follows. First, we present the theoretical background and explain the adopted framework for analysis of improvisation (Cunha et al., 1999, Kamoche et al., 2003). We then present our research approach and introduce *SmallSoft* and the process through which SPI technology was adopted over the period 1996-2005. Subsequently, we analyze the role of improvisation during the adoption process. Finally, we discuss the contributions of the research and the implications for SPI research and practice.

THEORETICAL BACKGROUND

Small software firms face special challenges when improving software practices. These challenges relate to the resources available and to having minimal influence over the environment. In the following, we review what is known about SPI in small software firms, and we present the improvisation framework that we have used as a lens to analyze the case.

Managing SPI Adoption

Managers in small software firms can find advice for SPI technology adoption in the literature. Most studies investigate small firms trying to adopt CMM or other rational SPI models. Typically, these studies describe the difficulties small software firms encounter and how these can be successfully resolved. Numerous studies suggest adaptations of CMM to fit small firms' needs (Kilpi, 1998, Kautz, 1999, Kelly and Culleton, 1999, Batista and Figueiredo, 2000, Horvat et al., 2000, Kautz et al., 2000, Casey and Richardson, 2004, Wilkie et al., 2005). Paulk argues that small firms' adoption of CMM "may be different in degree, but they are not different in kind" (Paulk, 1998) from those of other organizations. And, in general, it takes "professional judgment and understanding of how the CMM is structured to be used for different purposes" (Paulk, 1998).

Another group of mainly European studies has discarded CMM and developed alternative approaches fitted to the market dominated by small and middle-sized firms, but still keeping the basic values of the rational software organization intact. Examples are 3P approach (Brouse and Buys, 1999), IMPACT (Scott et al., 2001) Software Process Matrix (Richardson, 2001, Richardson, 2002), MESOPYME (Villalon et al., 2002) and COSMEA (Grechenig and Zuser, 2004, Steel, 2004).

This literature focuses on the downsizing of SPI technology to fit the resources of small firms, but do not address the culture differences, lack of knowledge, and external dynamic pressures that these firms experience.

The literature on SPI in small firms thus recommends downsizing or adapting the control centered, plan driven and rational improvement strategies when adopting SPI technology in small firms (Johnson and Brodman, 1997, Horvat et al., 2000, Leung and Yuen, 2001, Wilkie et al., 2005), without emphasizing the possible role played by unexpected events and initiatives outside managerial control. However, in our earlier study of SPI technology at *SmallSoft,* we found that the process fluctuated between management's attempt to control and events that caused the process to drift in unpredictable directions (Tjornehoj and Mathiassen, 2008). These findings suggest that everyday coping, bricolage, and improvisation play important roles when small software firms address their special challenges of adopting SPI technology (Ciborra, 2002).

Moreover, most studies of SPI literature are based on observations over limited time periods and focus on initial adoption of SPI technology. Only a few studies cover a considerable time span and report on how SPI initiatives evolve over time (Kuvaja et al., 1999, Balla et al., 2001, Richardson, 2001, Truffley et al., 2004). Thus, the literature provides little knowledge on how to achieve and sustain improvements over time in constantly changing environments.

In this study, we have therefore given center stage to understanding the role of improvisations in SPI technology adoption in *SmallSoft* over a period of ten years. To make sense of this adoption process, we focus on the interaction between events in the firm's environment and inside the firm, and we ask why, when, and how improvisation played a role in shaping the adoption process and outcomes.

Organizational Improvisation

Improvisation is a way of coping when time pressures, rapid shifting environments and lack of resources makes rational planning, decision processes, and knowledge creation difficult or even impossible (Ciborra, 1999, Cunha et al., 1999). Improvisation is often seen as the deviation from the norm of planning and rational decision-making; but increased uncertainty, complexity, and environmental dynamics create new conditions for firms in which the ability to improvise becomes more important (Chelariu et al., 2002). This is certainly also the case when it comes to technology adoption as described in the notion of drift (Ciborra, 2002, Ciborra and Willcocks, 2006, Elbanna, 2006, 2008). Emphasizing drift, the focus shifts towards "hosting technology" in the sense that a new technology is seen as a guest and that room is left for improvisation and mutual adaptation instead of merely focusing on diffusion (Brigham and Introna, 2006, Saccol and Reinhard, 2006).

Based on a comprehensive literature study, Cunha et al. (1999) suggest a definition of organizational improvisation:

> "… The conception of action as it unfolds, by an organization and/or its members, drawing on available material, cognitive, affective and social resources." (Cunha et al., 1999 p. 302)

The first part "The conception of action as it unfolds..." underpin that improvisation is basically characterized by timely convergence of planning and action and that improvisation is deliberate, extemporaneous and occurs during action. The second part "…drawing on available material, cognitive, affective and social resources" relates improvisation to bricolage (Cunha et al., 1999) by emphasizing that planning and action need to take place within the limits of available resources and knowledge. Additionally, the concept of "organizational" refers to when one or more individuals on behalf of some organizational unit takes action.

Improvisation can happen when:

> "… both (1) a demand for (a) speed and (b) action, and (2) an unexpected (and unplanned for) occurrence are perceived by the organization." (Cunha et al., 1999 p. 314)

Unexpected occurrences triggering improvisations can be experienced problems as well as perceived opportunities to take advantage of internal or external changes. However, improvisations will only take place if the organization offers a facilitating environment. An experimental culture, minimal structure, and a low procedural memory are important conditions for improvisation in organizations (Cunha et al., 1999). An experimental culture values action and experimentation when trying to understand and deal with the day-to-day reality in organizations. By minimal structure is meant a minimal control structure that facilitates focusing, coordinating, and keeping the necessary feeling of urgency, still leaving room for participants to innovate. Milestones and goal setting are recommended as efficient tools for this. Procedural memory is the amount of routine knowledge that organizations possess. If procedural memory is low, it leaves more room for improvisation since more events are unplanned. On the other hand, a high procedural memory, perceived as adaptable knowledge instead of unbreakable rules, will also enhance improvisation.

There are different degrees of improvisation, from minor variations over some known theme to full fleshed improvisations, and they also vary in type between process and product improvisations. How improvisations actually turn out is influences by leadership, member characteristic, information flow, memory related factors, organizational configurations and resources (Cunha et al., 1999). Improvisation can have negative as well as positive outcomes. Positive outcomes include motivation, flexibility, increased ability to improvise, new knowledge, as well as new routines and practices. Organizations also risk negative outcomes including inappropriate learning biased by actual circumstances, opportunity traps by not acquiring new knowledge, over amplifying emergent events, and

addictiveness to improvisation thereby under-utilizing existing knowledge and skills. In addition, there is the risk of increased anxiety and uncertainty for employees (Cunha et al., 1999).

Our analytical framework for investigating improvisation is summarized in Table 1. The framework is inspired by Cunha et al.'s (1999) improvisation concepts and the subsequent elaboration by Kamoche et al. (2003). In this framework, improvisation is triggered by unexpected, unplanned for events as organizational actors feel a need for immediate action that is within their action span. If this situation emerges and the appropriate conditions (experimental culture, minimal structure, low procedural memory) are in place, then improvisation can occur. The resulting improvisation is then analyzed in terms of the actions and their convergence with planning, the utilized material and resources, the degree, type, and influencing quality factors, and finally the outcomes of the improvisation. This framework is appropriate in the context of this study since it is centred round events that spark action and, as such, fits well with our event-based analysis of SPI technology adoption at *SmallSoft* (Newman and Robey, 1992, Peterson, 1998, Cho et al., 2008).

Construct	*Questions*
Trigger	Is the triggering event unexpected and unplanned for? Is the triggering event perceived as important? Do actors feel a need for immediate action within their action span?
Conditions	Are the conditions for improvisation in the organization apparent? • Experimental culture, minimal structure, procedural memory
Improvisation	Who acted and what did they do? How was planning and execution of action done? Which material and resources were utilized? What is the degree and type of the improvisation? Which quality factors influenced the improvisation and how? • Leadership, member characteristics, information flow, memory-related factors, organizational configuration, resources
Outcomes	What were the outcomes? • Positive: Increased flexibility, learning, emotional, further motivation • Negative: Biased learning, opportunity traps, over amplifying emergent events, addictiveness to improvisation, increased anxiety

Table 1 Improvisational framework inspired by (Cunha et al., 1999) and (Kamoche et al., 2003)

RESEARCH APPROACH

Longitudinal Interpretative Case Study

We have framed the investigation as a longitudinal, interpretative case study (Pettigrew, 1990, Walsham, 1993, 1995) based on a ten year-long collaboration with *SmallSoft* through a university-industry network (Mathiassen, 2002). This approach allowed us to analyze in detail how adoption of SPI technology evolved over time and to investigate the role played by improvisation in that process (Cunha et al., 1999, Kamoche et al., 2003).

SmallSoft is a small Danish IT-firm with approximately 50 employees. Their core competence is to combine domain specific engineering knowledge with IT-competencies to serve their customers by developing new software solutions or by adapting standard software. *SmallSoft* was founded when a big engineering-firm closed in 1987. Three engineers formed a consultancy firm operating in the same industry segment. They soon began to develop dedicated software tools in-house to support consultancy, which led them to develop a material-management-system as a joint venture with a main customer. Even though the virtue of this software was its built-in domain knowledge, not its technical quality, it soon evolved into an important standard product, and the initial ad-hoc development process laid the basis for the software practices of today.

In the early nineties, software production was established as a separate business area and *SmallSoft* started to employ additional developers, and also established a new department for tailored IT-systems. After a few years, the IT-business area dominated the firm that now had grown to 42 developers: 25 developers worked in the Standard Department responsible for the original and still important standard software product that over the years had developed into a portfolio of subsystems and versions; 15 developers worked in the Tailored Department adapting systems to a variety of customers; finally, a new department developed a storage management product based on a recent acquisition of a two person firm.

Under the impression of the bankruptcy, the founder's formed a firm culture during the first difficult years based on a low risk attitude and a belief in core engineering skills and long term personal customer relations. Despite growth, *SmallSoft*'s top management and board continued to exercise a defensive business and investment strategy, expecting a surplus every month. Employees were valued as innovative domain-knowledgeable experts, often working alone or in very small teams in close connection with customers. Software engineering skills were generally considered less important. During the mid nineties, the development of the standard product was fleshed out from development of customer versions to overcome severe quality problems. Within this subgroup of developers, software skills, software processes, and product quality became increasingly important.

Data Collection

Data were collected covering a period of ten years where we periodically worked with *SmallSoft* as employees, university teachers, and researchers engaged in action research (Susman and Evered, 1978, McKay and Marshall, 2001, Mathiassen, 2002). Our relationship with the firm began in 1999 when the first author was employed as project leader participating in implementing the new quality assurance system (QA-system - launched in 1996) and ended in 2005 when an action research project was concluded.

Through the activities, it was possible to collect a diverse and rich selection of data from a variety of sources covering a relatively long period of time; First, as part of a university-based action learning program, a group of employees at *SmallSoft* engaged in an SPI initiative starting in 2000, which was supervised by the second author. The resulting report included assessments of maturity, analysis of practice, descriptions of interventions, and a new SPI organization. Second, in an interview in March 2004, the key manager at *SmallSoft* described and reflected on important events in the firm's quality assurance (QA) initiative from 1996 to 2004. Third, in Fall 2005 we interviewed other key actors on their views of SPI technology adoption. The fourth main source was the first author's participation in an action research project with *SmallSoft* from Fall 2003 to Fall 2005; this initiative provided e-mail correspondence, documents, notes, minutes from meetings, and extensive research notes. Adding flesh to these primary data sources, we also had access to the firm's internal SPI document archive, graduate student reports from collaboration with the firm, recordings from meetings and work situations inside the firm, and interviews with research colleagues. Finally, we had extensive experiences from our personal participation in the process of adopting SPI technology at *SmallSoft*.

The direct participation in the two action research projects introduces possible bias in selecting and interpreting data. Also, the use of retrospective data from the interviews, although quite common (e.g., Denis et al., 1995, Sutton and Hargadon, 1996), involves the risk that interviewees forget or rationalize what originally happened. To reduce the adverse effect of these factors, we took advantage of our access to many different complementary sources to triangulate findings (Yin, 1994).

Data Analysis

Initially, we formed a historical map covering a time span of ten years (1996-2005). We viewed change processes as sequences of events, classified as either encounters or episodes (Newman and Robey, 1992). Episodes are relatively stable periods of evolution that are punctuated (Peterson, 1998) by compact periods of revolutionary events called encounters. Working systematically through the data sources, we identified candidate encounters by selecting events that were either mentioned in the interviews as important or that we found to have significantly impacted the adoption process. As an initial step towards a coherent case story, we mapped the identified encounters according to the chronological timeline and described them briefly together with the intermediary episodes. Subsequently, we went deeper into the data and started to analyze each encounter-episode. In doing so, we focused on SPI related activities, on their impacts on software development practices, and on activities and events in *SmallSoft*'s environment relevant to adoption of SPI technology. As these more detailed analyses progressed, we iterated the selection and description of encounters and episodes until a satisfactory story emerged.

To make sense of agency and the role of improvisation during SPI technology adoption at *SmallSoft,* we identified two intrinsically related socio-technical networks. First, there was the relatively stable and powerful *production*-network in which managers and software developers across *SmallSoft*'s three departments developed new solutions in response to customer requests, primarily based on knowledge and experiences available within the firm. Second, there was the less stable and weaker *improvement*-network through which a small group of different actors over time attempted to improve practices in the production-network through explication and diffusion of practices and adoption of new development technologies. These two networks offer complementary perspectives on *SmallSoft*'s adoption of SPI technology. One focuses on coping with the everyday reality of software production and the other on the ongoing SPI efforts. In line with Actor Network Theory (Callon, 1986, Latour, 1987, Callon and Law, 1989, Law, 1991, Walsham, 1997), each network includes actors within the firm, ways of working, current and emerging technologies, as well as relationships to forces outside the firm.

In the following, we first present a chronology of the encounters and episodes at *SmallSoft* based on our first analysis of the firm's adoption of SPI technology (Tjornehoj and Mathiassen, 2008). Subsequently, we present our analysis of these events through the lens of organizational improvisation (Cunha et al., 1999, Kamoche et al., 2003) based on the framework in Table 1. This analysis reveals how the adoption process was shaped through a combination of management control and improvisations triggered by unpredictable events in *SmallSoft*'s environment.

THE ADOPTION PROCESS

The adoption of SPI technology in *SmallSoft* passed through seven encounters and subsequent episodes during the period from 1996 to 2005. This chronology is summarized in Table 2.

	SPI-activity	SPI-organization	Immediate impact	Subsequent impact
ISO-9001 Certification (1996-1998)	Designing ISO-9001 QA-system	An ad-hoc representative QA-group (i.e. two employees and the QA-coordinator)	Successful ISO-9000 certification and implementation of new configuration management tool	Integration of QA-system into practice fails. Sustained use of configuration management tool
SPI-Action Learning (Spring 2000 to Jan. 2001)	Participating in action learning program in project management and SPI	An ad-hoc action learning group (three key employees). QA-coordinator is internal sponsor	Successful learning as new knowledge and energy was feed into the firm. No change of practices	Assessments planned and possible improvements identified
SPI-Pilot Projects (Autumn 2000 to Mar. 2001)	Assessing processes, planning and designing new process, conducting SPI-pilots	An ad-hoc action learning group (three key employees). Two production projects as pilots. Top management and supervisor in steering-committee	Successful CMM-assessment. Knowledge of appropriate change-practice gained. New processes designed and tested	New knowledge of SPI and organizational change acquired. Sporadic quality review practices
Forming SPI-Organization (Jan. 2001 to June 2001)	Designing, negotiating, and deciding on new centralized SPI-organization	A new centralized SPI-group appointed (two from action learning group and the QA-coordinator). Ad-hoc PIT-groups for each SPI-initiative	Successful design and decision of a new SPI –organization. Members appointed to SPI-group that started working	No change of practice. Successful evaluation of organizational change documented
SPI-Champion Exit (Nov. 2002 to Spring 2004)	Participating in SPI-action research project. Forced to focus on sales activities ended the SPI-activities	Until breakdown: The same central SPI-group with action researchers as consultants. After breakdown: No SPI-organization	The SPI-champion left the firm, partly because of the breakdown of SPI-initiatives	All SPI-initiatives stop
Learning from SPI-Failure (Summer 2004 to Summer 2005)	Management realizing SPI-breakdown and starting to reflect on present and past improvement practices	No SPI-organization. QA-coordinator is still personally engaged	QA-coordinator develops new understanding of SPI-practice. Renewed collaboration with researchers	Ongoing reflections on ideas and research to improve SPI efforts
A Grassroots Approach (Nov. 2005)	Designing and implementing a SPI PIT-organization	A permanent PIT-organization implemented. All employees participate in improvement team. QA-coordinator is sponsor	Successful and enthusiastic design and implementation of new grassroots SPI-organization	Three PITs (process improvement team) working. Eighteen employees and managers involved in improvement-network. Future success unpredictable

Table 2 Chronology of SPI technology adoption

ISO-9001-Certification

The QA-effort was initiated in 1996 and successfully completed in 1998 with an ISO-9001 certificate. *SmallSoft*'s QA-system was developed by a small improvement-network consisting of a department manager operating as QA-

coordinator and two key developers representing both management interests and engineering practices. The QA-coordinator's interest was to adopt best practices into the production-network as mandatory procedures in an online QA-system. No permanent QA-group or other quality techniques were implemented. Project managers were simply expected to comply with the QA-system after some introductory training.

After the certification, management left no doubt about the order of priority in *SmallSoft* as successful sales resulted in increased workloads; everyone focused entirely on producing software and serving customers. Only a new configuration management tool was successfully implemented and lead to improved practices and better customer relations. The tool was initially developed by employees in immediate response to customer dissatisfaction.

SPI Action Learning

In spring 2000, the local university offered *SmallSoft* an action learning education program (Mathiassen et al., 1999) in project management and improvement. *SmallSoft's* management welcomed this unexpected opportunity for an update on software engineering, for growing an informal university-industry-network within the region, and for boosting *SmallSoft*'s image as a professional firm. As a result, they formed a self-governing action learning group with three key employees that eventually developed a new improvement-network.

The action learning was conducted as university courses combined with supervised projects applying theory to support interventions in each participating firm. In their initial analysis, the action learning group identified difficulties within *SmallSoft* in prioritizing improvement in relation to the interests of the dominating production-network as the main threat to successful SPI.

SPI Pilot Projects

The action learning group agreed with management to conduct interventions according to the IDEAL-model (McFeeley, 1996). *SmallSoft* failed their internal assessment focused on level two: "the managed process". The lowest scores were found in the areas "project tracking and oversight" and "quality assurance" even though both had been the focus of improvement efforts before. Interventions into two pilot projects focusing on these two areas were planned to include four meetings with each pilot, two of which prepared the intervention while two practiced the new processes under supervision of the action-learning group. The introductory discussion of assessment results and the pilots participation in planning of the interventions created a positive attitude, but applying the newly designed procedures and templates for project planning was a failure, simply because lack of coordination with the stronger production-network regarding timing and alignment with customer needs. In contrast, the introduction of a new concept of quality meetings that was invented spontaneously was successful. The concept was

introduced into the production-network on a workshop and most developers welcomed the new quality process. Quality meetings were held in both pilots leading to improvements in plans and products.

Forming the SPI Organization

While engaged in the pilots, the action learning group negotiated a plan for continued SPI with management who gradually adopted SPI as a way to promote *SmallSoft* as a professional software house. Management was introduced to SPI by the university supervisor in October 2000 and approved a rather traditional proposal for strengthening the improvement-network in *SmallSoft* based on CMM in beginning of March 2001 (see Figure 1).

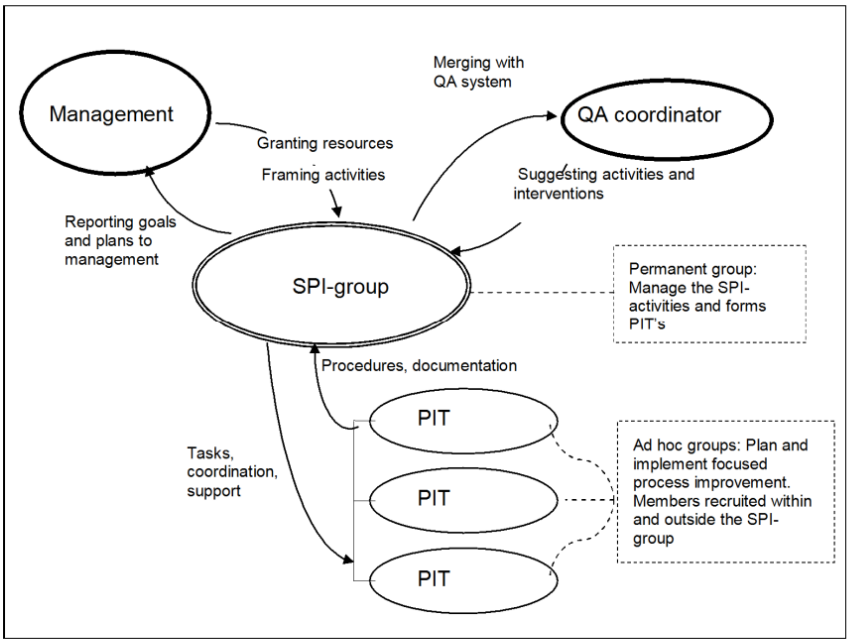

Figure 1 SPI organization based on CMM

The first task for the SPI group was to revise and implement the practices tested in the pilots, to prepare a new CMM assessment, and to evaluate existing SPI initiatives.

The action learning group had recorded 25 existing improvement initiatives that were either planned or so called bubblers – improvements spontaneously initiated within a department as part of the production-network. The bubblers were a well known part of the *SmallSoft* culture, as employees were expected to do their best and change practices if needed. This high degree of delegation led, over time, to important improvements of software practices. Management was very supportive of the new SPI organization, but the formation of the SPI group resulted in some

experienced developers feeling undercut in their professional authority since they had not been chosen as members.

All members of the SPI group were deeply involved in the production-network and could hardly find time or energy for SPI, delaying the introduction of the SPI organization from April to November 2001. Meanwhile, one member acted as SPI champion; he deeply believed in the quality meeting concept as a realistic way to improve practices and products. He pushed the SPI group, detailed and revised the quality meeting process and persuaded the others to implement the concept in their departments straight away. He personally conducted the first quality meeting in September 2001, but thereafter the SPI champion had less success engaging the other members of the SPI group.

SPI Champion Exit

Early 2003, *SmallSoft* entered an action research project with the local university hoping to revitalize SPI technology adoption. The SPI champion hoped the collaboration could bring about real change and dedicated personal energy and working hours, but the new-planned SPI activities were overwhelming for the SPI group.

During fall 2003, *SmallSoft* experienced a decline in the market, intensified their sales efforts and had to downsize for the first time ever in spring 2004. The situation drained focus away from the renewed SPI effort and *SmallSoft* postponed or cancelled most activities. The SPI champion became increasingly frustrated and finally left the firm. As a result, all SPI activities stopped.

Learning from SPI Failure

The failure of the centralized, norm-based SPI initiative forced the QA-coordinator to reflect on the situation. At this point, in fall 2004, he defined the central organization as a complete failure. He pointed to three reasons for the failure: lack of time in the SPI group, neglect from everybody else, and the need to maintain monthly economic surplus for the firm limiting investments in innovation. He realized the dominance of the production-network over the improvement-network, and reflected on how successful improvements had been driven by production needs, thus creating overlap between the production-network and the improvement-network. Several such improvements (bubblers) had been implemented during the relatively short lifetime of the firm, especially before the formation of the SPI group.

The QA-coordinator started to advocate for a grassroots-view of SPI as suitable for *SmallSoft*. Three considerations led to his conclusion: small firms cannot afford to invest in a dedicated improvement unit; the centralized SPI initiative implied increasing overhead; and, a centralized SPI group wouldn't know established practices well enough to propose realistic and efficient changes.

A Grassroots Approach

Projects and individuals of *SmallSoft* had proven capable of improving practices according to changes in demands, but exploration of grassroots approaches to SPI raised the question of how local improvements spread firm wide. In December 2004, the researchers conducted a social network analysis of the improvement-network displaying very close connections within each department, but only loose connections across department borders and to top management (Nielsen and Tjørnehøj, 2005). Thus, *SmallSoft* had a highly developed network for local improvements, but no basis for sharing knowledge and practices across departments.

These insights strongly impacted a SPI strategy meeting in March 2005 by engaging two department managers (including the QA-coordinator) and top management in a discussion of experiences with SPI, the QA-system, and the problems and benefits of having self-governed developers and departments. In the light of a welcomed sales boom making it difficult for the new storage management department to keep up with demands, management felt they could learn from the two established departments to avoid well known failures. Thus, knowledge sharing between departments had high priority within *SmallSoft* and management was at the meeting looking for a solution. Inspired by experience from another well-reputed Danish firm's success with adoption of CMM by involving all employees in process improvement teams (PITs), a new matrix-like SPI organization was immediately sketched.

Employees were assigned during summer 2005 and in August the PIT-organization was launched at a kickoff meeting. Each PIT focused on an improvement area and was initiated by management according to priority. In November 2005 three out of eight PITs were active, but working at a slower pace than expected. Eighteen developers and parts of management were now participating in an improvement-network that seemed to grow stronger than ever during adoption of SPI technology at *SmallSoft*.

IMPROVISATIONAL ANALYSIS

Considering these encounters and episodes of the adoption process and the underlying data from *SmallSoft*, we looked for signs of improvisational behavior. As illustrated by the example in Appendix A, we systematically analyzed each instance of improvisation following the framework in Table 1 (Cunha et al., 1999 p. 318, Kamoche et al., 2003). This approach helped us understand the role of improvisation during adoption of SPI technology at *SmallSoft.*

The 'ISO-9001-Certification' encounter focuses on evolving the QA-system for software development in *SmallSoft* and it is from an improvisational perspective an attempt to introduce more formalized procedural memory (Moorman and Miner, 1998). The reasons for introducing additional procedural memory into *SmallSoft* were partly trends within the software discipline, but mostly management's

intention to increase control over the production-network. The culture of the production-network was dominated by autonomous domain-experts with close customer contacts, mostly working in small one or two person projects. Because of the close connections to customers, these employees were constantly exposed to unexpected events as well as to time and performance pressures. They also generally lacked theoretical software engineering knowledge and only rarely shared experiences with other employees. These triggers, especially the feeling of urgency and the lack of formal or informal procedural memory, created a perceived need to act immediately on events, even though there might have been time to plan, search for knowledge, and carry out more structured actions. As a consequence, improvisations took place on a daily basis at the individual level of the production-network as responses to emerging project challenges utilizing personal skills and readily available knowledge. These improvisations were mentioned in both management and employee interviews as an intrinsic and successful part of *SmallSoft*'s culture.

The attempt to introduce a shared procedural memory was intended to reduce these improvisations by implementing procedures that were more appropriate in terms of efficiency, risk, and productivity (cf., the QA-coordinators interest in the QA-system). However the 'ISO-9001-Certification' episode was characterized by increasing time pressures due to a welcomed, but unexpected sales success. This exacerbated the feeling of urgency among members of the production-network and the level of improvisation was increased, rather than reduced, as people strived to satisfy emerging customer demands. Management accepted that the new procedures were pushed into the background, thereby sanctioning contradictory actions compared to the intentions behind the ISO-9001-Certification. One outcome of the ISO-9001-Certification was, however, very successful: the introduction and development of a configuration management tool across the production-network. The tool was initiated and developed with available resources (the development platform and knowledge within the firm) as an individual improvisation in response to dissatisfied customers within a project.

Focusing on the three encounter-episodes that started with the 'SPI Action Learning' encounter (see Table 2), different improvisations occurred related to the improvement-network on the organizational level. At this point, *SmallSoft* received an unexpected offer to join a local action learning effort. Management felt a need to exploit this opportunity thus improvising by asking three leading employees to participate without explicitly re-thinking the adopted improvement strategy. Instead, they expected the participants would bring back and utilize new knowledge and hence strengthen the improvement-network. The outcomes of this improvisation on SPI strategy were the centralized SPI organization and learning about organizational changes and software processes. The educational environment (including goal setting and exams) and the theories taught and shared offered a useful minimalist structure, and external leadership was provided by the university supervisor who

actively supported *SmallSoft*'s management. Another important outcome was increased flexibility and managerial openness towards the environment of the firm, based on the experience of successfully exploiting external resources through improvisations rather than learning from planned, long-term collaborations. At the operational level, between the three employees and the supervisor, the initiative represented novel approaches to improvement based on systematic learning and thorough planning. The outcome, the new SPI organization (Figure 1), was a success in relation to *SmallSoft*'s immediate goals. Therefore, it was unexpected that the central SPI organization never worked in relation to the day-to-day practices of the production-network. Even though this particular improvisation was guided by both appropriate minimal structural and leadership, the outcome did not fit the reality of *SmallSoft*.

During the subsequent breakdown in SPI technology adoption, where the SPI champion exited and the QA-coordinator reflected on the adoption of SPI technology, no improvisations took place related to the improvement-network. All available resources were spent on sales activities in response to the problematic economic situation at *SmallSoft*. However, as a result of the reflections, the QA-coordinator recognized the daily improvisations within the production-network had been a main source of improvement. Moreover, he feared that enforcing a procedural memory and building a strong, central SPI organization could weaken this productive, improvisational culture. In the encounter 'A Grassroots Approach', he therefore joined forces with the researchers and tried to find ways to merge the grassroots approach to SPI with the production-network, while at the same time meeting the need for knowledge-sharing and procedural memory.

When the new storage management department unexpectedly experienced increasing sales, the challenge related to knowledge transfer and sharing of software processes across *SmallSoft* was brought to the forefront of attention and this triggered a managerial improvisation leading to the PIT organization. The SPI manager called for a management meeting about lessons from the SPI research project, thereby creating a structure in which decisions could emerge. The researchers' presentation of the knowledge-sharing problems triggered a feeling of urgency among the participating managers. Inspired by how another well reputed firm had done SPI driven by PIT's, the new SPI organization was immediately sketched on the whiteboard as a variation over that practice. One important additional outcome was renewed and shared motivation among management to strengthen the improvement-network.

DISCUSSION

Improvisation at *SmallSoft*

Through the considered ten-year period, *SmallSoft* constantly improvised to improve their software operation in response to mainly external events. Being a

small firm in a turbulent environment and with a market under transformation (Mathiassen and Vainio, 2007), many events could not be foreseen and planned for, but they triggered action. Because the level of both formal and informal procedural memory at *SmallSoft* was low and the firm culture emphasized local experimentation and problem solving, improvisations played an important role during adoption of SPI technology (Moorman and Miner, 1998).

We found a high level of improvisations at the individual level as responses to locally experienced triggers and leading to improved software practices within specific projects in the production-network. These micro-level improvisations were characterized by impulsiveness, they utilized locally available resources and materials, and they were hardly supported by organizational structures and leadership. The micro-level improvisations were mostly uncoordinated with the overall goals of *SmallSoft*, highly situated, and rarely explicitly shared across projects and departments. However, we identified outcomes of micro-level improvisations that eventually grew to become important for the whole firm (e.g. the configuration management tool).

The micro-level improvisations dominated the production-network culture and ensured flexibility, personal learning, and motivated employees (Cunha et al., 1999 p. 327, Kamoche et al., 2003 p. 2030), but sometimes at the cost of over-improvising because of addictiveness to this form of response to external events (Miner et al., 1997, Cunha et al., 1999 p.332). There was a clear tendency within *SmallSoft* to fall back on this improvisational practice when pressures on the production-network unexpectedly increased, even in situations where other options were available (e.g. after the sales success when the QA procedures were pushed aside). The benefit of the micro-level improvisations was, however, that employees continued to produce what was expected, despite insufficient resources and a constantly changing environment. As a result, the micro-level improvisations at *SmallSoft* were expressions of an experimental culture with low procedural memory (Moorman and Miner, 1998, Cunha et al., 1999 p. 318); but, they were also the result of management practices that provided insufficient structural support, coordination, and leadership for improvisation in the production-network (Cunha et al., 1999).

We also identified important macro-level improvisations when management or management-related actors faced unexpected events related to *SmallSoft*'s goals and market position (e.g. the alternation of improvement strategy, when *SmallSoft* was offered the action learning). Macro-level improvisations unfolded at the organizational level with economy, strategy, reputation, and political factors playing important roles. Compared to the micro-level improvisations, it typically took longer time for actions to unfold and converge, but the triggering events were still characterized by being unplanned for and making people feel a need for immediate action. For this type of improvisation, we found examples of both high and low

levels of supporting structures and leadership (Brown and Eisenhardt, 1997, Cunha et al., 1999 p. 320). One instance (the SPI organization) suggests that high structure and leadership did not necessarily ensure effective macro-level improvisations; in fact, structures, coordination mechanisms and leadership were in this case too dominant and distant from the everyday reality of the production-network within *SmallSoft*. Another example (the PIT-organization), suggests that it was possible to facilitate improvisations with innovative outcomes by deliberately arranging for a situation with sufficient, but minimal structure and leadership as well as the right people with few, but focused resources.

In summary, the inviting culture and the longstanding tradition for local, in-flight problem solving helped *SmallSoft* improvise in response to a turbulent and unpredictable environment. For improvisations to be useful they had to address appropriate challenges (e.g. the configuration management tool and the new PIT-organization) and be supported and coordinated in ways that would ensure benefits for the firm. However, improvising within *SmallSoft* when there was no need (e.g. the micro improvisations, when there were actually time to plan, search for knowledge and carry out more structured actions) - because events could be planned for through procedural memory (routine) or because there was no real time or performance pressure - jeopardized rather than enhanced efficiency and effectiveness in *SmallSoft*'s production-network.

Improvisation in SPI

The *SmallSoft* case documents the important, but not unproblematic role improvisations can play in helping small software firms cope with the challenges related to SPI technology adoption. Planned centralized adoption of SPI technology (Humphrey, 1989) driven mainly by rational models (Paulk et al., 1993, Koch et al., 1994) may introduce initiatives that are poorly aligned with the reality of the firm (Aaen, 2003). At *SmallSoft,* this happened during both the SPI Action Learning and the SPI Pilot Projects initiatives. In contrast, the outcomes of improvisations are shaped by the immediate challenge of the triggering event and the resources at hand - hence facilitating that SPI technology is adapted to the reality of the firm (Ciborra, 2002). At *SmallSoft,* this was illustrated by the bubblers or micro-improvisations in the production-network and the invention of the PIT-organization as part of the improvement-network. Improvisations like these engage a firm in actively hosting a new technology (Ciborra, 2002) by iteratively adapting the technology to the evolving everyday life within the firm. This understanding is in starch contrast to the mainstream SPI literature, where the adaptation of SPI models are seen as the result of deliberate and centralized actions (Humphrey, 1989, Aaen et al., 2001, Aaen, 2003).

Bricolage constitutes an important aspect of improvisation (Cunha et al., 1999) allowing for the unthinkable to happen as actors combine outcomes and solutions in new and innovative ways (Ciborra, 2002) hence overcoming the constrained

resources that hinder SPI technology adoption in many small firms (Villalon et al., 2002). One significant example of bricolage at *SmallSoft* is the invention of the configuration management system by the end of the ISO-9001 initiative. The system was designed and programmed in-house by one programmer driven by customer complaints, it was based on readily available knowledge and tools, and, it ended up as a firm wide backbone for management in the production-network.

Since planning converges with action in improvisations, actors are empowered and gain motivation from seeing immediate progress (Barrett, 1998, Cunha et al., 1999 p. 327, Kamoche et al., 2003), in contrast to the situation in traditional SPI efforts where it normally takes several complex and expensive initiatives to reach new maturity levels (Aaen et al., 2001). The momentum gained from improvisations in *SmallSoft* is especially visible in the micro-level improvisations before the efforts were organized as a centralized SPI initiative (Figure 1). Many SPI efforts stops after initial planning and assessments and (Herbsleb and Goldenson, 1996), as a consequence, they do not bring changes to the firm . Empowering and motivating participants through improvisational action may help avoid this unfortunate situation.

An important challenge in facilitating SPI technology adoption through improvisation is to ensure coordination and alignment of initiatives and outcomes with firm goals (Cunha et al., 1999 p. 320, Vera and Crossan, 2004). In traditional SPI, coordination and alignment is ensured through centralization and by relying on comprehensive models in identifying and prioritizing improvement initiatives (Humphrey, 1989). In *SmallSoft,* the centralized approach stopped all or most improvisational SPI activity and it did not lead to other SPI activities suggesting that alternative or complementary means of coordination and alignment are required in small software firms. Leadership can be executed differently depending on the type of improvisation (Kamoche et al., 2003), but it plays a key role in helping improvisations become useful for the organization as a whole. Coordination of improvisations can be based on minimalist structures to ensure a shared feeling of urgency and an appropriate level of coherence among individual actors. Without implying specific outcomes, third-order controls, shared goal-setting, and deadlines are useful examples of such minimalist structures (Cunha et al., 1999). While such structures were only sparsely present in relation to micro-level improvisations at *SmallSoft*, deliberately choosing and utilizing them as part of management practices could have increased the benefits of improvisations without restraining possible outcomes.

Another challenge is to avoid over-amplifying the impact of emergent events and getting addicted to improvisation (Miner et al., 1997, Cunha et al., 1999 p. 332). In *SmallSoft*, this was a real challenge because of lack of procedural memory, weak knowledge sharing, and established management practices, and also because people appeared to enjoy coping with difficulties and developing individual responses. In

contrast, in traditional SPI best practices are defined and spread throughout the organization to increase efficiency and predictability (Humphrey, 1989). This raises the delicate issue of balancing the amount of formal procedural memory with improvisations to achieve appropriate levels of efficiency and effectiveness while at the same time facilitating innovative solutions and widespread motivation and empowerment. One important answer may well lay in consistently viewing procedural memory as adaptable knowledge instead of unbreakable rules (Cunha et al., 1999, Mathiassen and Pourkomeylian, 2003).

The literature has a strong emphasis on planned, centralized adaptation of existing SPI models (Humphrey, 1988, Rout, 1995, McFeeley, 1996, Aaen et al., 2001) and on critiquing these models (Bach, 1994, Nielsen and Nørbjerg, 2001, Trienekens et al., 2001, Conradi and Fuggetta, 2002, Aaen, 2003) and documenting failures to adopt SPI technology. However, there has been little emphasis on considering improvisations as a complementary approach to SPI technology adoption that can help firms take advantage of the potential benefits without risking failure. The *SmallSoft* case shows in detail how improvisational actions can help overcome the weaknesses of traditional SPI approaches. At the same time, the case uncovers important risks that have to be handled as firms exploit improvisations during SPI technology adoption.

Implications for Management

These insights from *SmallSoft* combined with theory on organizational improvisation, suggest specific lessons for managers of small software firms that want to adopt complex technologies like SPI.

Cultivate improvisations. The experimental culture at *SmallSoft* facilitated improvisation and sustained the innovative capability of the firm. In some situations, however, people improvised when they did not have to. To help small software firms stay innovative, it is important to cultivate knowledge, values, and practices that enable improvisational capability (Barrett, 1998, Cunha et al., 1999 p. 318, Vera and Crossan, 2004). At the same time, it is important to set standards and secure sharing of procedural memory to avoid improvising in routine situations or situations that can actually be planned for because such behaviors easily lead to inefficient work practices and low quality (Moorman and Miner, 1998, Cunha et al., 1999 p. 321, Kamoche et al., 2003) .

Facilitate deliberate improvisations. While micro-level improvisations into the production-network of *SmallSoft* were rather impulsive, macro-level improvisations into the improvement-network were more deliberate. Unplanned events were evaluated to understand the potential of SPI for the firm, and, based on this evaluation management deliberately decided whether to improvise. We advise managers to identify and analyze unexpected events in the environment to determine whether they are suitable triggers for deliberate improvisation into the

firm's improvement-network to facilitate technology adoption (Cunha et al., 1999, Kamoche et al., 2003, Vera and Crossan, 2004).

Provide support structures. The improvisation during the 'Action learning' and 'SPI pilots' encounters, see Table 2, was guided by explicit coordination mechanisms supported by the minimal structures of the educational initiative. The improvisation thus became manageable and coherent, aiming at a well known and shared goal that included both learning and change. Even though the product, i.e., the formalized SPI organization, was rather unsuccessful, the outcome was mostly positive in terms of learning, flexibility, and motivation. Coordination can be provided formally, but often explicitly shared goal setting (sometimes involving management) and deadlines are sufficient (Cunha et al., 1999). In adoption processes, the technology can also provide important minimal structure and serve as an implicit coordination mechanism (Kamoche et al., 2003). Management is advised to consider appropriate coordination mechanisms and structures to facilitate improvisations (Barrett, 1998, Cunha et al., 1999 p. 320). Management should also ensure that structures stay minimal to leave as much room for improvising as possible (Cunha et al., 1999, Kamoche et al., 2003).

Exercise leadership. More explicit leadership could have sparked further innovation within *SmallSoft.* The recommended leadership style is supportive and collective turn taking, inviting and encouraging each member to contribute so relevant knowledge, skills, and resources add to useful outcomes (Barrett, 1998, Cunha et al., 1999 p. 321). Management of small software firms are advised to exercise such leadership that will reduce pressures and create room for individual and group improvisations related to technology adoption. Such leadership should also communicate organizational objectives through clear goals and other minimal structures to facilitate appropriate coordination of improvisational outcomes (Vera and Crossan, 2004).

CONCLUSION

Adoption of complex technologies like SPI typically fluctuates between management's attempt to control the adoption process and events that causes the process to drift in unpredictable directions (Ciborra, 2002, Tjornehoj and Mathiassen, 2008). To further understand how drifting occur and contribute to shaping the adoption process, we investigated the role of improvisation in a longitudinal study of SPI technology adoption in *SmallSoft* covering a ten year period. We found that improvisations were enabled by a deeply rooted experimental culture as *SmallSoft* responded to a turbulent environment during the adoption process. Two different forms of improvisations interacted, often in uncoordinated ways, to shape innovation within the firm. Micro-level improvisations were mostly individual, they typically targeted the production-network, and they were shaped through personal practices and skills when people faced new and unexpected project challenges. Macro-level improvisations were more deliberate, they typically

targeted the improvement-network, and they involved managerial attempts to make actions converge on the organizational level. The uncoordinated interactions between these two forms of improvisations played a major role during adoption of SPI technology at *SmallSoft*.

We combined the empirical insights from *SmallSoft* with theoretical insights from the literature on organizational improvisation to provide implications for management of small software firms. However, the major insights from the study are specific to the context at *SmallSoft* and we therefore encourage more research on the role of improvisations during adoption of complex technologies in small software firms. Our study suggests that improvisations can be beneficial for small firms to enhance employee creativity, motivation and empowerment, to create momentum in the adoption process despite constrained resources, and, most importantly, to help effectively adapt the technology to the everyday circumstances of the firm. However, to avoid the risks of improvising in routine situations and of generating outcomes that are uncoordinated with firm goals, appropriate leadership and minimal structure is needed. To better understand the opportunities and risks involved, more research is needed on the role of improvisation in different contexts of technology adoption. Also, the general literature on organizational improvisation focuses mainly on the micro-level and downplays interactions with macro-level improvisations over time and across levels of analysis. Building on the findings from *SmallSoft,* we therefore encourage further research on the organizational interplay between different and sometimes opposing forms of improvisation over time.

REFERENCES

Bach, J. (1994). The Immaturity of the CMM, *American Programmer* 7(9): 13-18.

Balla, K., Bemelmans, T., Kusters, R. and Trienekens, J. (2001). Quality through managed improvement and measurement (QMIM): Towards a phased development and implementation of a quality management system for a software company, *Software Quality Journal* 9(3): 177-193.

Barrett, F. J. (1998). Creativity and Improvisation in Jazz and Organizations: Implications for Organizational Learning., *Organization Science* 9(5): 605.

Batista, J. and Figueiredo, A. D. D. (2000). SPI in a Very Small Team: a Case with CMM, *Software Process: Improvement and Practice* 5(4): 243-250.

Brigham, M. and Introna, L. D. (2006). Hospitality, improvisation and Gestell: a phenomenology of mobile information, *Journal of Information Technology* 21: 140-153.

Brodman, J. G. and Johnson, D. L. (1994). What Small Businesses and Small Organizations say about the CMM, in 16th International Conference on Software Engineering (ICSE-16) (Sorrento, Italy, 1994); IEEE Computer Society Press. 331-340.

Brouse, P. S. and Buys, R. T. (1999). Affordable ways to improve application development, *IT Professional* 1(4): 47-52.

Brown, S. L. and Eisenhardt, K. M. (1997). The Art of Continuous Change: Linking Complexity Theory and Time-Paced Evolution in Relentlessly Shifting Organizations, *Administrative Science Quarterly* 42(1): 1-34.
Callon, M. (1986). Some elements of a sociology of translation: domestication of scallops and the fishermen of St Brieuc Bay, in Law, J., (Ed.) *Power, action and belief: a new sociology of knowledge?* London: Routledge & Kegan Paul, pp. 196-223.
Callon, M. and Law, J. (1989). On the Construction of Socio Technical Networks: Content and Context Revisited, *Knowledge and Society* 9(8): 57-83.
(2005). Process Maturity Profile (http://www.sei.cmu.edu/appraisal-program/profile/pdf/SW-CMM/2006marSwCMM.pdf), http://www.sei.cmu.edu/appraisal-program/profile/pdf/SW-CMM/2006marSwCMM.pdf
Casey, V. and Richardson, I. (2004). A practical application of the IDEAL model, *Software Process: Improvement and Practice* 9(3): 123-132.
Chelariu, C., Johnston, W. J. and Young, L. (2002). Learning to improvise, improvising to learn: A process of responding to complex environments, *Journal of Business Research* 55(2): 141-147.
Cho, S., Mathiassen, L. and Nilsson, A. (2008). Contextual dynamics during health information systems implementation: an event-based actor-network approach, *European Journal of Information Systems* 17: 614-630.
Ciborra, C. (1999). Notes on improvisations and time in organizations, *Accounting, Management and Information Technologies* 9: 77-94.
Ciborra, C. (2002). The Labyrinths of Information: Challenging the Wisdom of Systems, New York: Oxford University Press.
Ciborra, C. and Willcocks, L. (2006). The mind or the heart? it depends on the (definition of) situation, *Journal of Information Technology* 21: 129-139.
Conradi, R. and Fuggetta, A. (2002). Improving Software Process Improvement, *IEEE Software* 19(4): 92.
Cunha, M. P., Cunha, J. V. and Kamoche, K. (1999). Organizational improvisation: what, when, how and why, *International Journal of Management Reviews* 1(3): 299-341.
Dion, R. (1992). Elements of a Process-Improvement Program, *IEEE Software* 9(4): 83-85.
Elbanna, A. R. (2006). The validity of the improvisation argument in the implementation of rigid technology: the case of ERP systems, *Journal of Information Technology* 21: 165-175.
Elbanna, A. R. (2008). Strategic systems implementation: diffusion through drift, *Journal of Information Technology* 23(2): 89-96.
Grechenig, T. and Zuser, W. (2004). Creating Organic Software Maturity Attitudes (COSMA) Selected Principles and Activities for Software Maturity in Small and Medium Software Enterprises, in Fourth International Conference on Quality Software (QSIC'04) (Braunschweig, Germany, 2004); 134-143.

Hansen, B., Rose, J. and Tjørnehøj, G. (2004). Prescription, description, reflection: the shape of the software process improvement field, *International Journal of Information Management* 24(6): 457-472.
Herbsleb, J. D. and Goldenson, D. R. (1996). A systematic survey of CMM experience and results, in the 18th International Conference on Software Engineering (ICSE-18) (Berlin, Germany, 1996); IEEE Computer Society Press. 323-330.
Horvat, R. V., Rozman, I. and Györkös, J. (2000). Managing the complexity of SPI in small companies, *Software Process: Improvement and Practice* 5(1): 45-54.
Humphrey, W. (1989). Managing the Software Process, Reading, MA: Addison-Wesley.
Humphrey, W. S. (1988). Characterizing the Software Process, *IEEE Software* 5(2): 73-79.
Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991). Software Process Improvement at Hughes Aircraft, *IEEE Software* 8(4): 11-23.
Johnson, D. L. and Brodman, J. G. (1997). Tailoring the CMM for small businesses, small organizations, and small projects., *Software Process Newsletter* 8: 1-6.
Kamoche, K., Cunha, M. P. E. and Cunha, J. V. D. (2003). Towards a Theory of Organizational Improvisation: Looking Beyond the Jazz Metaphor, *Journal of Management Studies* 40(8): 2023-2051.
Kautz, K. (1999). Making sense of measurement for small organizations, *IEEE Software* 16(2): 14-20.
Kautz, K., Hansen, H. W. and Thaysen, K. (2000). Applying and Adjusting a Software Process Improvement Model in Practice: The Use of the IDEAL Model in a Small Software Enterprise, in 22nd International Conference on Software Engineering (Limerick, Ireland, 2000); ACM Press. 626-633.
Kelly, D. P. and Culleton, B. (1999). Process Improvement for Small Organizations, *Computer* 32(10): 41-47.
Kilpi, T. (1998). Product Management Challenge to Software Change Process: Preliminary Results from Three SMEs Experiment, *Software Process: Improvement and Practice* 3(3): 165-175.
Koch, P., Kuvaja, S., Mila, L., Krzanik, A., Bicego, S. and Saukkonen, G. (1994). Software Process Assessment and Improvement: The BOOTSTRAP Approach, Oxford: Blackwell Publishers.
Kuvaja, P., Palo, J. and Bicego, A. (1999). TAPISTRY - A software process improvement approach tailored for small enterprises, *Software Quality Journal* 8(2): 149-156.
Latour, B. (1987). Science in Action: How to follow Scientists and Engineers through Society, Cambridge, MA: Harvard University Press.
Law, J. (1991). Introduction: Monsters, Machines, and Socio Technical Relations, in Law, J., (Ed.) *A Sociology of Monsters: Essays of Power, Technology and Domination,* London: Routledge.
Leung, H. K. N. and Yuen, T. C. F. (2001). A process framework for small projects, *Software Process: Improvement and Practice* 6(2): 67-83.

Mathiassen, L. (2002). Collaborative Practice Research, *Information, Technology & People* 15(4): 321-345.
Mathiassen, L., Borum, F. and Pedersen, J. S. (1999). Developing Managerial Skills in IT Organizations: A Case Study based on Action Learning, *Journal of Strategic Information Systems* 8(2): 209-225.
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. (Eds.) (2002). *Improving Software Organizations: From Principles to Practice,* Reading, MA, Addison-Wesley.
Mathiassen, L. and Vainio, A. M. (2007). Dynamic Capabilities in Small Software Firms: a Sense-and-Respond Approach, *IEEE Transactions on Engineering Management* 54(3): 522-538.
Mcfeeley, B. (1996). IDEAL: A User's Guide for Software Process Improvement, Pittsburgh, PA: Software Engineering Institute
Mckay, J. and Marshall, P. (2001). The Dual Imperatives of Action Research, *Information Technology & People* 14(1): 46 - 59.
Miner, A. S., Moorman, C. and Bassoff, P. (1997). Organizational improvisation in new product development: Marketing Science Institute, MSI.
Moorman, C. and Miner, A. S. (1998). Organizational improvisation and organizational memory, *The Academy of Management Review* 23(4): 698-723.
Newman, M. and Robey, D. (1992). A Social Process Model of User-Analyst Relationship, *MIS Quarterly* 16(2): 249-266.
Ngwenyama, O. and Nielsen, P. A. (2003). Competing Values in Software Process Improvement: An Assumption Analysis of CMM from an Organizational Culture Perspective, *IEEE Transactions on Engineering Management* 50(1): 100-112.
Nielsen, P. A. and Nørbjerg, J. (2001). Assessing Software Processes: Low Maturity or Sensible Practice? *Scandinavian Journal of Information Systems* 13: 51-68.
Nielsen, P. A. and Tjørnehøj, G. (2005). Mapping Social Networks in SPI, in IFIP 8.6 Conference: Business Agility and IT Diffusion (Atlanta, GA, USA, 2005); Springer Verlag, 2005.
Paulk, M. C. (1998). Using the Software CMM in small organizations, in The joint Pacific Northwest Software Quality Conference and the Eighth International Conference on Software Quality (Portland, 1998); 350-361.
Paulk, M. C., Curtis, B., Chrissis, M. B. and Weber, C. (1993). Capability Maturity Model for Software version 1.1, Pittsburgh, PA: Software Engineering Institute
Peterson, M. F. (1998). Embedded organizational events: Units of process in organization science, *Organization Science* 9: 16-33.
Pettigrew, A. M. (1990). Longitudinal Field Research on Change:Theory and Practice, *Organization Science* 1(3): 267-292.
Richardson, I. (2001). Software Process Matrix: A Small Company SPI Model, *Software Process: Improvement and Practice* 6(3): 157-165.

Richardson, I. (2002). SPI models: What characteristics are required for small software development companies? *Software Quality Journal* 10(2): 101-114.
Rodenbach, E., Van Latum, F. and Van Solingen, R. (2000). SPI - A Guarantee for Success? - A Reality Story from Industry, in Second International Conference, PROFES 2000 (Oulu, Finland, 2000); Springer. 216-231.
Rout, T. P. (1995). SPICE: A Framework for Software Process Assessment, *Software Process: Improvement and Practice* 1(1): 57-66.
Saccol, A. Z. and Reinhard, N. (2006). The Hospitality Metaphor as a Theoretical Lens for Understanding the ICT Adoption Process, *Journal of Information Technology* 21: 154-164.
Scott, L., Jeffery, R., Carvalho, L., D'ambra, J. and Rutherford, P. (2001). Practical Software Process Improvement -The IMPACT Project, in 13th Australian Software Engineering Conference (ASWEC'01) (Canberra, 2001); IEEE. 182.
Steel, A. P. C. (2004). Low-rigour, Rapid Software Process Assessments for Small Software Development Firms, in 2004 Australian Software Engineering Conference (ASWEC'04) (Melbourne, 2004); IEEE. 323-334.
Susman, G. I. and Evered, R. D. (1978). An Assessment of the Scientific Merits of Action Research, *Administrative Science Quarterly* 23(4): 582-603.
Saastamoinen, I. and Tukiainen, M. (2004). Software Process Improvement in Small and Medium Sized Software Enterprises in Eastern Finland: A State-of-the-Practice Study, in 11th European Conference on Software Process Improvement (EuroSPI) (Trondheim, Norway, 2004); Springer Verlag. 69-78.
The-Cmmi-Product-Team (2001). Capability Maturity Model Intergration (CMMI-SM), Version 1.1, Continuous Representation, Pittsburgh, PA: Carnegie Mellon, Software Engineering Institute
The-Cmmi-Product-Team (2002). CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1, Staged Representation (CMMI-SE/SW/IPPD/SS, V1.1, Staged), *Technical Report of SEI,* Pittsburgh, PA: SEI
Tjornehoj, G. and Mathiassen, L. (2008). Between control and drift: negotiating improvement in a small software firm, *Information Technology & People* 21(1): 69-90.
Trienekens, J., Kusters, R. and Van Solingen, R. (2001). Product focused software process improvement: Concepts and experiences from industry, *Software Quality Journal* 9(4): 269-281.
Truffley, A., Grove, B. and Mcnair, G. (2004). SPICE for Small Organizations, *Software Process: Improvement and Practice* 9(1): 23-31.
Vera, D. and Crossan, M. (2004). Theatrical Improvisation: Lessons for Organizations, *Organization Studies* 25(5): 727-749.
Villalon, J., Agustin, G. C., Gilabert, T. S. F., Seco, A. D., Sanchez, L. G. and Cota, M. P. (2002). Experiences in the Application of Software Process Improvement in SMES, *Software Quality Journal* 10(3): 261-273.
Walsham, G. (1993). Interpreting Information Systems in Organizations, Chichester: Wiley.

Walsham, G. (1995). Interpretive case studies in IS research: nature and method, *European Journal of Information Systems* 4(2): 74-81.
Walsham, G. (1997). Actor-network theory and IS research: current status and future prospects, in Lee, A. S., Liebenau, J. & Degross, J. I., (Eds.) *Information Systems and Qualitative Research,* London: Chapman and Hall, pp. 466-480.
Ward, R. P., Fayad, M. E. and Laitinen, M. (2001). Software Process Improvement in the Small - A Small Software Development Company's Most Difficult Challenge: Changing Processes to Match Changing Circumstances, *Communications of the ACM* 44(4): 105-107.
Wilkie, F. G., Mcfall, D. and Mccaffery, F. (2005). An Evaluation of CMMI Process Areas for Small- to Medium-Sized Software Development Organizations, *Software Process: Improvement and Practice* 10(2): 189-201.
Wohlwend, H. and Rosenbaum, S. (1994). Schlumberger's Software Improvement Program, *IEEE Transactions on Software Engineering* 20(11): 833-839.
Aaen, I. (2003). Software Process Improvement: Blueprints versus Recipes, *IEEE Software* 20(5): 86-93.
Aaen, I., Arendt, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement, *Scandinavian Journal of Information Systems* 13: 81-101.

Appendix A

Table 3 below illustrates how the framework for analysis of improvisation adapted from (Cunha et al., 1999) and (Kamoche et al., 2003) was utilized. The example covers the improvisation following the triggering event "unexpected increase in sales" during the encounter "ISO-9001-Certification".

Construct	***Questions***
Trigger	Is the triggering event unexpected and unplanned for? *Unexpected increase in sales lead to new customers (macro-level) and add-on in the existing projects (micro-level). SmallSoft had no plans for rapid growth in number of customers and in workload.* Is the triggering event perceived as important? *Management and employees felt that speedy action was needed for SmallSoft to stay in business (delayed or no delivery => no add on sales and bad reputation => out of business)* Do actors feel a need for immediate action within their action span? *Action possible (Within action span) - they could work more hours and improvise more.*
Conditions	Are the conditions for improvisation in the organization apparent? *In general SmallSoft had an extensive experimental culture and a tradition for very minimal structure because of high delegation of responsibilities for work. The level of procedural memory was still low, since the just designed QA-system had not been brought in action and was now further ignored.*
Improvisation	**Macro-level:** Who acted and what did they do? *Upsizing of the production network and ignoring the QA-system were the immediate actions from management leading to employees and increased service to customers (old and new), but also to ignorance of the new procedures* How was planning and execution of action done? *The actions were unplanned before the situation occurred and was acted out a new in every new situation in the episode* Which material and resources were utilized?

30

	Employees working overtime utilizing their existing knowledge and skills in coping with the unknown situation. Everybody using their professional networks (for upsizing) What is the degree and type of the improvisation? *Degree: Variations over known themes i.e. not interfering with the micro-level improvisations and every day coping of the production and letting go of all overhead actions* *Type: Mostly individual and behavioural but both aiming at product and processes* **Micro-level:** Who acted and what did they do? *Experienced employees responding to increased workload and new demands from customers and the upsizing (e.g., integrating new colleagues) in many ways. – not following procedural memory or preplanning, but still producing service, products and learning also for new employees. The products of the improvisation were service and deliveries to the customers and integration of the new employees* How was planning and execution of action done? *The employees were acting on the fly. Plans were at best very short term if at all present* Which material and resources were utilized? *Procedural memory (the new procedures) did not play a role. Employees working habits served as or instead of plans while utilizing their existing knowledge and skills in coping with the unknown situation* What is the degree and type of the improvisation? *Degree: Variations over known themes (working habits) -> some full fleshed improvisations, but in the small and individually and locally* *Type: Mostly individual and behavioural but both aiming at product and processes* **Both micro- and macro-level:** *Leadership was at micro-level mostly individual while at macro-level the leadership could bee seen in monthly follow up on economic status and deliverables underpinning the priority* *No significant membership characteristics since everybody was*

	part taking. *The information flow was low, since everybody focused narrowly on their immediate task leaving no time for information processing or sending* *Organisational memory did not bind the employees when improvising* *Organization Configuration enhanced improvisation through widespread trusting relationships between employees and through a decentralized and delegated organization of the work tasks leaving lots of room for improvising.*
Outcomes	What were the outcomes? Positive: *Increased flexibility: SmallSoft gained experience in quick upsizing. E.g., learned how to utilize professional networks, etc.* *Learning: New employees integrated and trained, experienced employees gathered more experience – especially in how to cope through improvising* *Emotional: Satisfaction from actually coping, feeling proud, feeling of belonging* *Further motivation: Through the positive emotions above and through the success.* Negative: *Biased learning: Since the QA-system was ignored, the organization learned to value it less and not utilize it when being busy.* *Opportunity traps: Addicted to individual improvisations* *Over amplifying emergent events: Maybe they over amplified the hastiness of the events – They could have negotiated more time for planning and delivery, so they could have worked according to procedural memory at least with existing customers.*

Table 3 Improvisation following "unexpected increase in sales" during "ISO-9001-Certification"

D. Social Network Analysis in Software Process Improvement

SOFTWARE PROCESS IMPROVEMENT AND PRACTICE
Softw. Process Improve. Pract. (2009)
Published online in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/spip.419

Social Networks in Software Process Improvement

Peter Axel Nielsen* and Gitte Tjørnehøj
Department of Computer Science, Aalborg University, Selma Lagerlöfs Vej 300, Aalborg East DK-9220, Denmark

Research Article

Software process improvement in small organisation is often problematic and communication and knowledge sharing is more informal. To improve software processes we need to understand how they communicate and share knowledge. In this article have studied the company SmallSoft through action research. In the action research we have applied the framework of social network analysis and we show this can be used to understand the underlying structures of communication and knowledge sharing between software developers and managers. We show in detail how the analysis can be done and how the management can utilise the findings. From this we conclude that social network analysis was a useful framework together with accompanying tools and techniques. Copyright © 2009 John Wiley & Sons, Ltd.

KEY WORDS: software process improvement, social network analysis, communication, knowledge sharing, software management

1. INTRODUCTION

Software process improvement (SPI) has long been a concern for software companies. The development of the Capability Maturity Model (CMM) (Humphrey 1989, 1992, 2002) and later CMMI (Ahern *et al*. 2001, 2003; Chrissis *et al*. 2003) by the Software Engineering Institute sparked a huge interest in the field. The CMM family of models has also been supplemented with the IDEAL approach (McFeeley 1996) that addresses how to utilise the CMM.

The focus in this paper is on SPI in small and medium-sized companies. A core characteristic of small software companies seems to be that they often face changing environments and are more vulnerable than large companies. Ward suggests, 'the processes by which software is developed are likely to change with circumstances – perhaps even change dramatically – even while general principles like the need for good communication remain constant' (Ward *et al*. 2001).

Within SPI there is an explicit concern that the maturity-model approaches, like CMM and CMMI, are not adequate for improving small software companies. An early survey raises the concern that the CMM does not fit small software companies (Brodman and Johnson 1994). Several studies of SPI for small companies reveal many difficulties: small companies cannot necessarily afford the investment in SPI (Kautz and Larsen 1997); small companies lack SPI knowledge (Cater-Steel 2001); they see SPI as bureaucratic (Kelly and Culleton 1999); and they see traditional SPI methods as too costly (Villalon *et al*. 2002).

The difficulties for small software companies cannot be attributed to the CMM-based approaches alone as there are reported examples of successful SPI, some which are CMM-based and some which are not. Kautz *et al*. describe a successful improvement effort of a small software company, in which CMM was used for the initial maturity assessment and IDEAL was used to structure the improvement effort (Kautz and Thaysen 2001; Kautz *et al*. 2001).

* Correspondence to: Peter Axel Nielsen, Aalborg University, Computer Science, Aalborg, Denmark
† E-mail: pan@cs.aau.dk

Also (Kelly and Culleton 1999) reports on attempts to develop and test new SPI approaches for small software companies based on CMM. In yet another study it is suggested that while CMM is used for assessment, it is necessary to supplement it with what the authors call an 'action package concept' to overcome small companies' lack of follow-through into action planning and implementation (Villalon *et al*. 2002).

There are reports of successful SPI where CMM or other maturity models were not used. Kautz has studied process improvement in three small companies (Kautz 2000, 1999). The success of SPI in these companies is attributed to four factors (Kautz 2000): a tailored approach; an experience network between companies; external assistance; and partial external funding. In another study a medium-sized company's problems with current software processes were assessed with a technique for problem diagnosis, which was not on the basis of a maturity model and many of the identified problems could later be alleviated (Iversen *et al*. 1999; Nielsen *et al*. 2002). In this study the success of the improvement effort was attributed to the particular way experience and knowledge were shared during the problem diagnosis.

Sharing knowledge, also sometimes referred to as sharing experience, is fundamental in all these reports. Hence, we have undertaken research to understand knowledge sharing better and in greater detail. We report on an action research effort in the software company SmallSoft on how knowledge sharing can be understood through social network analysis and how software managers can utilise social network analysis to manage SPI efforts.

The current theories behind knowledge sharing in SPI and social network analysis are presented in more detail in section 2. In section 3, we outline our research approach and describe the data collection and data analysis. In section 4, we present the case company SmallSoft and how we used social network analysis in that company. In section 5 we discuss the role of social network analysis in SPI in general and for SmallSoft in particular. The article concludes in section 6.

2. KNOWLEDGE SHARING AND SOCIAL NETWORK ANALYSIS

It appears that part of the success of SPI in small companies has to do with how knowledge is shared among the developers and managers partaking in the SPI effort. Knowledge management is a relevant perspective to apply in SPI in general. Kautz and Thaysen concur with this and put forward that knowledge in SPI is not only to be seen as a simple commodity, but needs to be understood in a much broader and social context (Kautz and Thaysen 2001). Understanding knowledge management is a key to SPI according to other studies (Baskerville and Pries-Heje 1999; Mathiassen and Pourkomeylian 2003; Pries-Heje and Pourkomeylian 2004).

There are several reasons why knowledge management is an important perspective. First, it has been established that software development depends hugely on communities-of-practice, which differ from the formal organisation (Mathiassen 1998, p. 88). Communities-of-practice create the specific context as well as the shared experience and understanding of their members in such a way that they shape how new or modified processes are adapted, implemented or rejected.

Second, SPI is a problem-solving activity (Mathiassen *et al*. 2002, p. 4) where problems in software processes have to be identified, needs have to be understood, possible improvements have to be devised and prioritised, and actions to improve must be taken. All these activities require communication of different perceptions and interests, of plans and priorities, and of outcomes.

Third, SPI is also a knowledge creating activity (Mathiassen *et al*. 2002, p. 7) where SPI knowledge needs to be elicited from experience, some experience has to be explicated, concerns for capture and quality of available knowledge have to be addressed, and validated feedback has to be provided.

Fourth, organisational influence processes are important in SPI (Nielsen and Ngwenyama 2002). This study of influence processes concludes that it is crucial to understand the networks through which power and influence is exercised; but also that a major source of power is knowledge and communication skills.

On this background we find it interesting to analyse the networks through which knowledge is shared and communicated in greater detail. We expect that it can advance SPI in general, and in small companies in particular. Social network analysis is a framework for such detailed analyses.

DOI: 10.1002/spip

Social network analysis is a general framework and a set of techniques applied to study the relationships between organisational actors and their exchange of resources; see (Wasserman and Faust 1994; Cross and Parker 2004). In social network analysis organisations are viewed as consisting of actors linked together in networks through action, exchange, and interpretation and sharing of resources like information and knowledge. Social network analysis seeks to provide a way to look at an informal organisation, which exists in parallel to the formal and hierarchical organisation chart. In this view, organisations are made up of interdependent actors with relational ties between them. Network models conceptualise structure as lasting patterns of such relational ties (Wasserman and Faust 1994). Wasserman and Faust further define actors as discrete individuals, or corporate or collective social units, (i.e. not only as a single person). The relational ties can be of varying types: evaluation of one person by another (as with friendship), transfer of material resources, affiliation, and authority (as between managers and subordinates), and behavioural interaction like sending messages and engaging in a discussion (Wasserman and Faust 1994, p. 18).

Social network analysis is not a new approach. It has been developed and applied in a large number of organisational studies (see Tichy *et al*. 1979; Wasserman and Faust 1994; Scott 2000), but it has not been applied directly in SPI efforts before. Social network analysis has been applied to understand software teams (e.g., Yang and Tang 2004; Ehrlich *et al*. 2007; Long and Siau 2007; Müller *et al*. 2008). It has however mostly been applied to distributed software teams and to open source development.

The framework does not provide a unit of analysis and data may be collected about many different kinds of actors and relational ties. It is, however, common to collect data about the contents of the relational ties as well as their intensity and reciprocity. On the basis of the collected data, the approach requires the study of network properties and structural characteristics. Some of the properties that we will also investigate later in this study are the following:

- *Density:* how well-connected are the network's actors?
- *Centrality:* who is the 'most important' actor in a network?

A network can be analysed for these properties, but these are just a few of the analyses that can be performed on a social network. The analyses all have a foundation in graph theory (Borgatti and Everett 1992; Wasserman and Faust 1994; Scott 2000), but the interpretation and the semantic implication of these analyses remain specific to the setting where the data were collected.

Our application of social network analysis focuses on the social networks through which software process improvement may happen and in particular we focus on communication about SPI as a means for sharing knowledge.

3. RESEARCH APPROACH

The research followed collaborative practice research (CPR) that is an action research approach (Mathiassen 2002). The CPR approach builds on (Checkland 1991; McKay and Marshall 2001) and it guides how interventions into software companies' practice can take place and how scholarly knowledge is gained. Mathiassen argues that CPR is suited for research into: (i) how SPI activity may be understood through practice studies, (ii) how support for SPI activity may be developed, and (iii) how interventions may improve SPI activity.

For this purpose the two authors were part of a SPI group in the company SmallSoft over several months; not on a daily basis, but sufficiently often to get a good understanding of the company and its SPI activity.

The data collection and data analysis for the action research study was performed in two parts. For the purpose of understanding the software company and its context the researchers collected data about: (i) SmallSoft's background, (ii) its SPI activity, (iii) its history with SPI, and in particular (iv) all minutes from meetings in the SPI group, and (v) progress reports. The data were analysed informally to inform the researchers and to write the case background in section 4.

For the purpose of taking action informed by social network analysis the researchers collected and analysed the data following a more stringent procedure. The procedure is similar to that outlined by (Cross and Parker 2004, p. 143), and it contains the following steps:

1. Identify the group
2. Collect data about relationships

DOI: 10.1002/spip

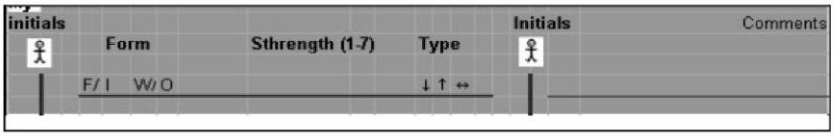

Figure 1. Graphical questionnaire

3. Visually analyse the results
4. Feedback the results to the group and validate the results
5. Evaluate the outcome

As we wanted to investigate to what extent and in which ways the company communicated and shared knowledge about software process improvement we identified the relevant group as all developers and all managers in SmallSoft.

Each respondent was asked to fill in a graphical questionnaire. They were asked to assess their communication on issues of improving software processes in the company during the last six months. They were asked to identify and characterise the communication as they recalled it. For each interaction they were asked to provide the name(s) of their colleagues in the interaction and to assess whether the communication had been (see Figure 1):

- Formal or informal by circling 'F' or 'I'.
- Written or oral by circling 'W' or 'O'.
- Downward, upward, or lateral influence process by circling one of the three arrows.
- Strength indicated by a number from 1 to 7; 1 meaning very low (e.g. receiving an email) and 7 meaning very high (e.g. collaboration or continuous dialogue).

The graphical survey questionnaire is shown in Figure 1. The accompanying instruction told the respondent to also register the initials of their communication partners and use a new line for every interaction. The instrument provides this pattern for all interactions.

This means that every reported line in a returned questionnaire is evidence of a relationship.

All questionnaire data were transferred directly to a spreadsheet. The format used in the spreadsheet was then loaded into NetDraw[1], which is a tool for social network analysis that can display graphs with actors as nodes and relationships as edges. Both nodes and edges can have attributes.

The tool offers various display features and analyses, which are performed automatically by the built-in graph algorithms. The tool was used to analyse and keep an overview of the data using graphical elements to visualise structures in the social networks (e.g. to select parts of the graph, show different attributes and weights, identify central actor, cut-point). The tool was also used to find and illustrate several network structures like centrality, components, k-cores, etc. We explain these concepts in more detail in subsection 4.2 where the actual analysis is described.

The analysis of the network data was iterative. The researchers were consistently looking for patterns in the network models, which confirmed or rejected working hypotheses about the company's SPI activities. That led to analytical insight, which in turn led the researchers to modified and new working hypotheses. The iterative analysis was temporarily stopped to validate the findings with two department managers. Their feedback was used to extend the iterative analysis. It also gave a detailed impression of which network models were relevant from a management point of view. The managers found that some of the models provided interesting research hypotheses and proper findings, but were not providing valuable managerial insight. The managers' feedback also led the researchers to prompt several developers to respond to the questionnaire to increase the data coverage.

The analysis ended with a second session with all three managers (for a description of the case company see subsection 4.1). This second session later moved on to a presentation of the findings and a rather detailed discussion of what to do about the network problems and SPI (see section 4.5).

4. SOCIAL NETWORKS IN SPI

The modelling of social networks followed the approach outlined in section 3. In this section we

[1] NetDraw: www.analytictech.com/Netdraw/netdraw.htm.

DOI: 10.1002/spip

first present the case background, then we present the models and analyses, and finally we report from the validation and managerial utilisation of the findings.

4.1. Case Background

SmallSoft is a small software company with two departments. The ERP Department develops a large ERP system and maintains it at a number of customer sites. The department's tasks are characterised by long-term and close contacts with a few large customers. The software developers have much domain knowledge within logistics in their customers' area. The head of this department is also responsible for the quality system and the company's ISO9000 certificate. He was also heading the SPI group. The Tailor-Made Department develops several tailored systems for many different customers. Their products range from traditional administrative systems to web portals. The application domains vary and the developers' primary expertise lies within software engineering and project management.

Previously, improvements in SmallSoft's software development were casual and spread through collaboration and informal contacts between colleagues. A few significant improvements had attracted management's attention and were turned into company-wide improvements. One company-wide improvement led to an internal software development project, which produced a support tool for tracking development tasks. Most improvements, however, were small and remained personal or local among a few colleagues.

When the research began, the company was introduced to a basic SPI approach and soon top management announced the slogan 'CMM level three – in three years.' A SPI group was formed and a developer from each of the departments was appointed to the group. The group took on the responsibility of assessing the current practices, planning improvement initiatives, and implementing these. Successful improvements were supposed to be added to the existing quality system. The manager of the ERP department later characterised this new set-up as a failure. His perception was that some developers felt pushed aside and that others stopped focusing on improvements waiting for the results from the SPI group. The SPI group on their part lacked time and resources and organised only one improvement initiative. At the same time the company experienced a market decline and subsequent low sales figures, and this led to a shift of focus away from improvement activities and towards sales activities and monthly sales figures.

Despite these setbacks, SmallSoft's management recognised the value of their previous improvements as vital for their business success and found it necessary to proceed. The two department managers' shared perception was that future improvements had to be rooted in a strategy that would provide faster feedback as well as visible and immediate benefits for the software developers. It was in this atmosphere that the analysis of social networks was initiated.

4.2. Social Network Analysis of SPI in SmallSoft

The analysis had the immediate purpose of understanding SmallSoft's social networks as a basis for managerial decisions about SPI. To that end we chose to visualize the models of communication networks that emerged from the data when displayed with NetDraw.

The most basic network model is shown in Figure 2. The node distribution feature in NetDraw provided its visual layout.

The model should be read in the following way. Circles represent developers; white circles denote that they are from the Tailor-Made department and grey circles show developers from the ERP department. Developers 29 and 30, depicted in black, are no longer in the company. Developers 4, 10–13 have not responded to the questionnaire and no others have reported communication with these developers. Triangles denote managers; manager 9 is the CEO. The number of respondents therefore is 23 of 28 staff, or 82%. When these models were used in SmallSoft the real names of developers and managers were shown.

The graph analyses follow Scott as well as Wasserman and Faust (1994); Scott (2000). The initial *graph analysis* was to look for components and central actors, because this provides a good overview to begin with. These first analyses have been performed on the network from Figure 2 where connections are un-directed and considered without their attributes and strengths.

The component analysis was on the basis of the formal concepts of component, cut-point, and

DOI: 10.1002/spip

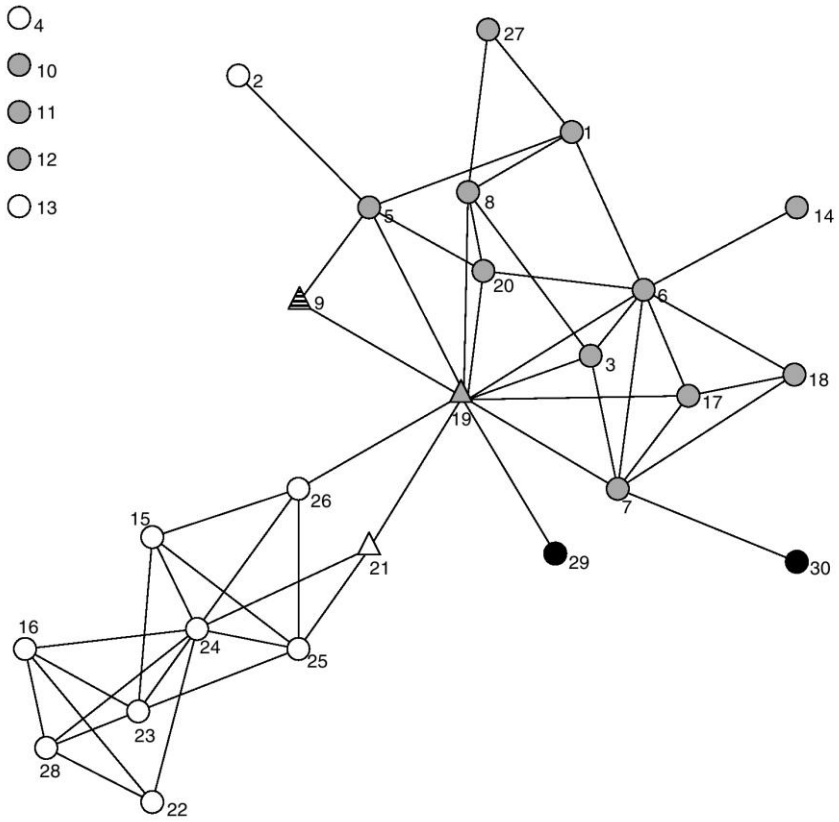

Figure 2. Basic network for communicating about SPI in SmallSoft

clique. A *component* in a social network is defined as a maximal connected subnet (Scott 2000, p. 101; Wasserman and Faust 1994, p. 109). The model of SmallSoft reveals that it consists of a single component because all developers and managers are related to at least one other except developers 4, 10–13 whom no one communicated with. These outliers as well as developers 29 and 30 were removed in subsequent analyses. A *cut-point* is a node whose removal would increase the number of components (Scott 2000, p. 107; Wasserman and Faust 1994, p. 112). In Figure 2, manager 19 is a cut-point who would split the company in two components, and developer 5 is a cut-point who would disconnect developer 2 from the main component. Similarly developer 6 is a cut-point who would disconnect developer 14. A *clique* is a subnet in which every possible pair of nodes is directly connected and the clique is not contained in any other clique (Scott 2000, p. 114–115; Wasserman and Faust 1994, p. 254). Counting only those subnets with more than three nodes the following subnets are cliques: (16, 22, 24, 28); (15, 24, 25, 26); (6, 7, 17, 18). Cliques are highly connected and as there are no cliques larger than four; they are mere small islands in SmallSoft.

The centrality analysis is on the basis of the formal concepts of degree centrality, closeness centrality, betweenness centrality and peak. The measure for *degree centrality* is defined as a node's number of direct relations (Scott 2000, p. 83 Wasserman and Faust 1994, p. 178). The more direct connections a node has the more central it is. The node with the highest degree centrality is: manager 19 with a degree of 10; see Table 1. The measure for *closeness*

DOI: 10.1002/spip

Table 1. Centrality measures

Rank	Degree	Closeness	Betweennes
1	M19 (10)	M19 (39)	M19 (144)
2	D24 (8)	D26 (44)	D24 (60)
3	D6 (8)	D21 (45)	D26 (57)
4	D7 (6)	D6 (48)	M21 (41)

centrality is defined as the sum of distance to all other nodes (Scott 2000, p. 86 Wasserman and Faust 1994, p. 184). The closer a node is to all other nodes, the more central it is; the closeness of nodes for all nodes in the SmallSoft network is as follows: manager 19 has the distance 39 to all other developers and managers in SmallSoft; see Table 1. The *betweenness centrality* of a node is defined as the proportion of node pairs that has the node on its path (Scott 2000, p. 87; Wasserman and Faust 1994, p. 191). NetDraw computed these to: manager 19 is on the path between 144 pairs; see Table 1. A *peak* is a node with a higher centrality measure than any other node which (s)he is directly connected to. Manager 19 is such a peak in all three centrality measures, while developer 24 is a peak on degree and betweenness centrality.

The qualitative analysis started with a working hypothesis that Smallsoft was a very informal organisation, but with strong monitoring by its two department managers. Furthermore, prior to the social network analysis the researchers held the perception that SmallSoft's management was in control and that all SPI activity had to be communicated through the managers. Consequently, the researchers did not assume that there were social subnets with the capacity nor the inclination to communicate independently on SPI and take action on SPI.

The subsequent analysis of the model led to the following results. As mentioned above, first of all it shows that five developers are completely outside all communication about SPI. Second, it identifies one main component containing both the ERP and Tailor-Made departments. Manager 19 is the most central actor as he is the actor ranking first on all three centrality measures. Manager 19 is also a peak as he is more central than any other actor he is connected to. This is not surprising as 19 is the manager of the ERP department and responsible for the quality system, the ISO9000 certificate, and also the SPI manager. He is connected to the top manager, CEO 9, and all connections between the ERP department and Tailor-Made department go through him.

Manager 21, the manager of the Tailor-Made department, is far less central and not a peak. He shares the linkage to the ERP department with developer 26. In the Tailor-Made department, developer 24 with a degree of 8 is the only peak and he is connected to everyone in the department. The path from any of the managers to any of their developers is less than or equal to two edges. In the ERP department this is due to the central role of the manager and in the Tailor-Made department it is due to developer 24.

Returning to the graph analysis for details we chose to go into depth with the cohesiveness of the company and the two main departments. To that end we modelled the k-cores network in NetDraw. A *k-core* is a maximal subnet in which each node is adjacent to at least *k* other nodes (Scott 2000, p. 110; Wasserman and Faust 1994, p. 266). The 3k-core displays the actors with a degree greater than or equal to three. Where the clique analysis shows the size of the most well-connected islands (none were larger than four), the k-core analysis shows the overall cohesiveness of SmallSoft.

The 3k-core model in Figure 3 is only slightly different from the basic model – only the CEO and three developers were removed. The 4k-core is much smaller as it removed five more developers. The 3k-core model shows the connectivity of the network and it is evident that the inner coherence of the company is relatively strong. It is noteworthy though that the CEO is not part of the inner network where SPI is addressed.

4.3. Analysis of the Relation Attributes

Figures 4 and 5 show the attributes of the communication for the main component. The four models in Figure 4 illustrate the differences between formal and informal communications and between written and oral communications. The communication is mostly informal and all actors are involved in informal communication. Formal communication is only found around the two peaks (manager 19 and developer 24) and between the two departments. Written communication has a stronger presence in the Tailor-Made department and around the manager of the ERP department. Oral communication is widespread and every actor participates in oral

DOI: 10.1002/spip

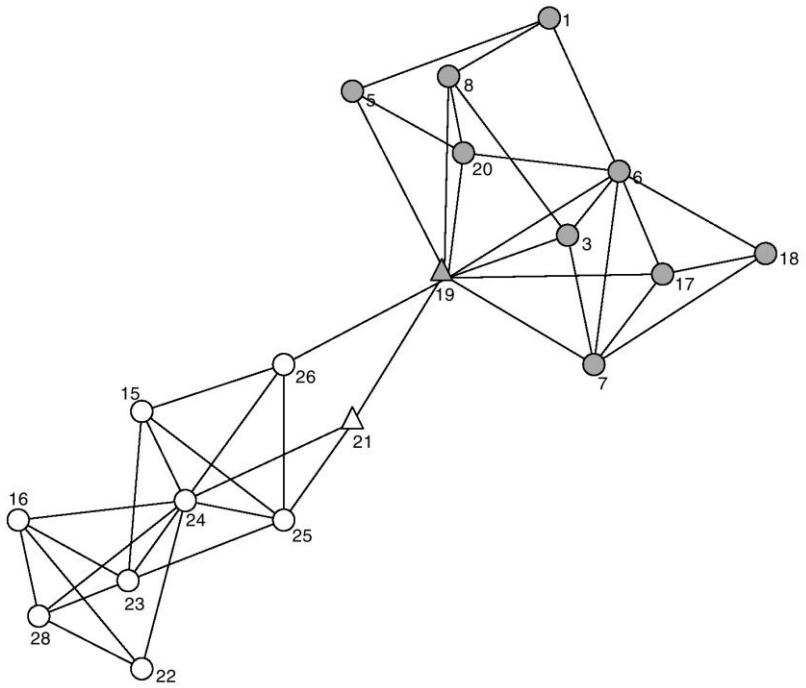

Figure 3. The 3k-core model for SmallSoft

communication. It is worth noticing that the communication between the departments is formal but oral.

The questionnaire asked all respondents to categorise the direction of the influence process in the communication as downward, upward or lateral. An influence process is an attempt by an originator to influence another individual or group to achieve goals (Nielsen and Ngwenyama 2002; Kotter 2003). Downward and upward influencing refers to relations in the formal hierarchy, while lateral refers to influencing between peers. Figure 5 shows all communication ties that have been reported as lateral by the respondents excluding all those reported as downward and upward. It is not surprising that the managers become isolated in the lateral network. What is interesting is that the lateral networks are present and involve all developers. What is also interesting, but not immediately apparent in Figure 5 is that there is communication among developers that has not been reported as lateral influencing. These are small in numbers, but they show that communication networks in SmallSoft are not entirely congruent with the hierarchy of formal authority.

Respondents had been asked to assess the strength of the communication on a scale from 1 to 7 with seven as the highest. Figure 6 shows a model of communication strength where respondents have reported strengths of 5–7 corresponding to the top half (not counting a strength of 4 which is the middle position). According to the weak-tie theory by Granovetter low strength is efficient for knowledge sharing because it bridges otherwise disconnected groups while high strength will lead to redundant information because group members know what the others know (Granovetter 1973). This has been further qualified in a more recent study in which it was shown that weak ties help search for useful knowledge, but transfer of complex knowledge requires strong ties (Hansen 1999).

Overall, the communication ties between the departments seem rather low. The model also shows that there are strong communication ties in the Tailor-Made department with degrees of 2 or more for most developers while the manager 21 has no

DOI: 10.1002/spip

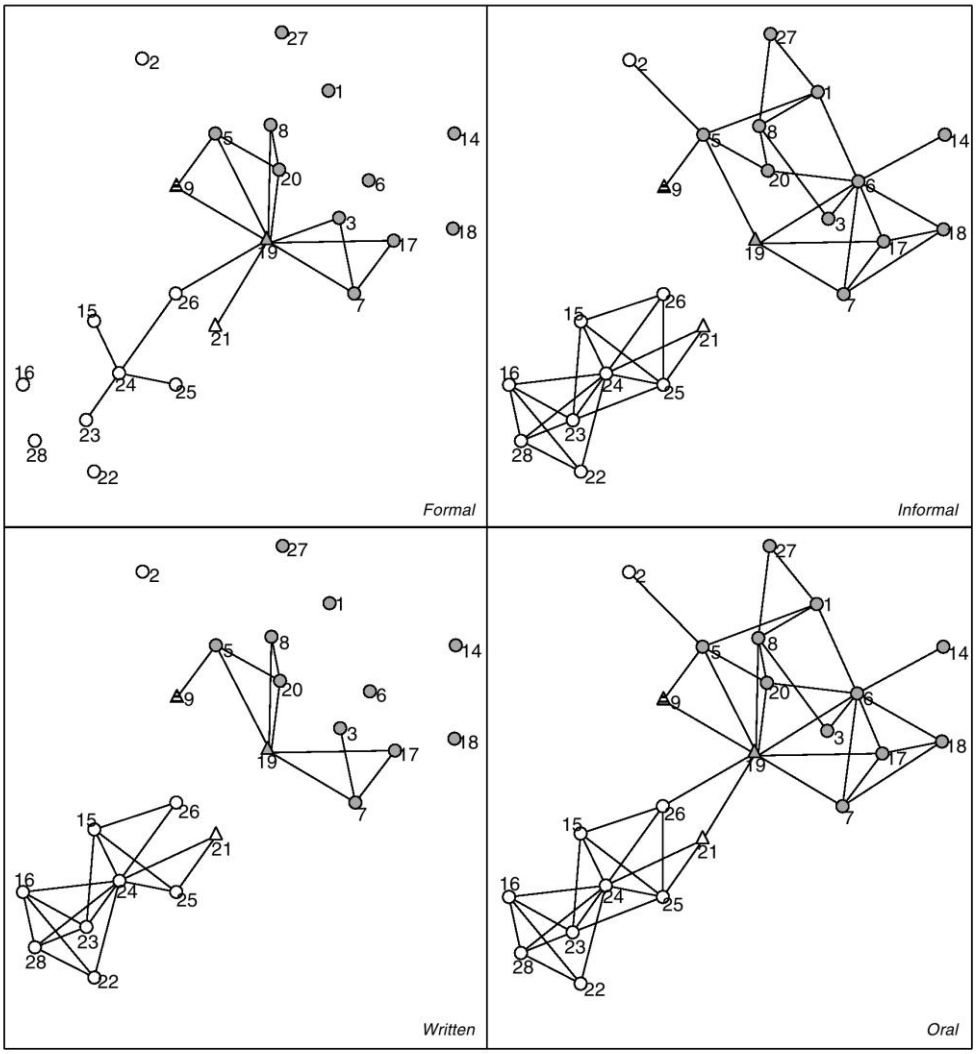

Figure 4. The models of attributes in the SmallSoft

strong communication with his own developers. For the ERP department the pattern is uneven as some are strongly connected while several are weakly connected. This would suggest on the basis of Hansen's theory that SmallSoft as a whole will have the social network to search for useful knowledge, but transfer of the more complex knowledge in software process improvement will happen within the departments.

4.4. Analysis Presented to Management

Overall the social networks show two departments with an informal, mostly oral and widespread interaction within the departments, but with sparse contact between departments and to top management. The ERP department has a central manager, 19, gate-keeping the department against all the other actors in the company in a more formal way than

DOI: 10.1002/spip

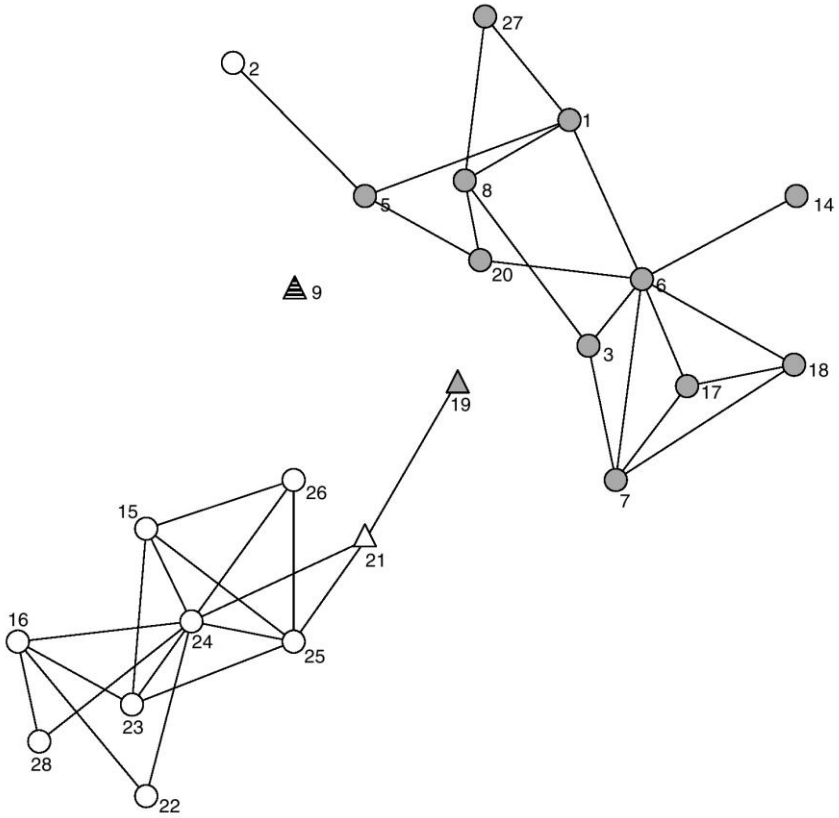

Figure 5. A model of lateral communication

usual in other parts of the company. He controls the communication on improvements both within his own department and at the management level. He is the only middle manager with contact to the top management.

The Tailor-Made department has a strong internal actor in developer 24 keeping the department connected and communicating intensively with many other developers. The manager of the department, 21, plays a lesser role in SPI as he has fewer ties and partakes only in lightweight communication. He only connects to the whole department through developers 24 and 25. This looks like a widespread delegation of responsibility for SPI.

Until the time of the analysis SmallSoft had followed a centralised and formal improvement strategy. There are considerable misfits between a centralised strategy and the underlying social networks. This may largely explain the failure of the improvement effort so far. The underlying social networks are uneven. In the ERP department, developers are unaccustomed to written communication. In both departments the sub networks are also lateral and thus less disposed to acting on formal management directive. In contrast, the applied centralised SPI strategy is management-driven and communicated in formal writing. The social network analysis thus leads to the conclusion that either the social networks must change or another strategy must be chosen. Social networks are emergent and cannot easily (if at all) be designed and it is thus more appropriate to change the strategy.

Faced with these alternatives Smallsoft's management wants to change to a decentralised strategy. They assess that this will suit the company better

DOI: 10.1002/spip

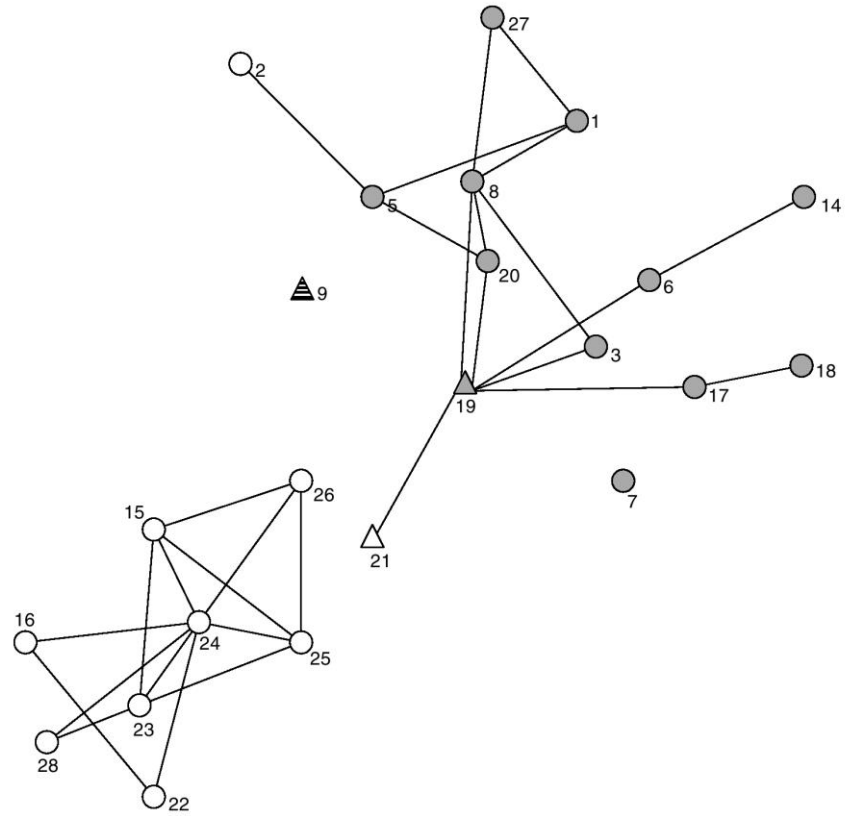

Figure 6. A model of strengths in communication; s ≥5

and will involve more developers. When embarking on a decentralised SPI strategy the researchers analysis led to the following:

- The remarkably weak ties between the two departments certainly hinder a central and cross-departmental SPI approach also in the future.
- A serious management commitment to SPI will be very difficult to exercise with so little communication on SPI involving the top manager; perhaps the lack of management involvement shows that SPI is not of strategic importance to the company's business strategy.
- Few improvements will spread easily from one department to the other. Closer ties need to be built between the two departments and at the level of the developers. If this is impossible or undesirable, the departments should be seen as separate social networks and independent SPI activities should be organised in each department deliberately decreasing dependency on cross-department knowledge sharing.
- Any SPI initiative in SmallSoft will benefit from at stronger collaboration among the managers and also involving the CEO.
- The ERP department could benefit from decentralisation, less formalisation and delegation of responsibilities. Manager 19 could very well be overloaded with responsibilities. If that is the case, he is a bottleneck that inhibits improvements and hinders knowledge sharing and communication in the department. Management commitment to SPI is on the basis of real involvement and focus.
- The network structures uncovered by the analyses do not hinder ideas and improvements being communicated amongst developers.

DOI: 10.1002/spip

These advices for the SPI managers are very much in line with (Cross and Parker 2004). They suggest that it is a management task to initiate, develop and maintain networks. They further propose that the internal network structure of a company should be aligned with its environment. For a small company like SmallSoft the environment for the ERP department changes only slowly, but it is vulnerable to a few missed sales opportunities. In the Tailor-Made department there are often changes that it should respond to. Management should hence consider whether they want to move developers between the departments.

4.5. Management Reflections

The researchers presented the models and their analysis to the three managers, who then discussed the findings derived from the network models. The managers' understanding of the current situation differed between the two middle managers on the one hand and the CEO on the other hand. The two department managers, 19 and 21, saw the SPI activities from within, and the CEO observed SPI from outside. Not surprisingly, the CEO disagreed with the finding that he was marginal to SPI in the company. All three managers recognized the problematic situation with the loose coupling between the two main departments. Though being in favour of a decentralised SPI strategy, they agreed that reducing collaboration between the departments would increase business risk and that it would be too costly and inefficient if each department organised its own independent improvement activities. Thus they looked for a solution that would build closer ties between developers across departments to achieve easier diffusion of improvements through informal and oral communication, i.e. gradually cultivate and improve the underlying social networks. On the other hand, the new solution should provide management with sufficient overview and insight so that improvement activities could receive more management attention and be supported by more formal communication from the management.

In line with (Cross and Parker 2004, p. 91) the researchers suggested that the company should introduce particular teams responsible for each of their improvement areas. This advice suggests that building bridges between individuals and between subnets can improve network relations. Building bridges means among other things (a) initiating relations and (b) develop professional and personal relations.

This suggestion was not immediately approved and decided at the meeting between the researchers and managers, but within a few months a few new teams were organised across the departments. They involved software developers with special interest in each improvement area. Management assigned 20–40 working hours per month to each team to support the activities. Like all other projects in the company the teams had to report their work and progress to management through monthly reports.

To support and coordinate the improvement teams a coordinating SPI group was formed. The new improvement initiative was kicked-off at a meeting for all teams where some of the social networks were presented to explain management's reasons for establishing the new teams.

5. DISCUSSION

The most significant finding of the social network analysis for SmallSoft was that there was already communication about SPI and that knowledge about software development was already shared. The managers already knew this in general, but they did not know the details. The network models showed many details which the managers were unaware of and which they had not addressed in their dealing with software process improvement.

The social network analyses proved valuable in SmallSoft. They provided the researchers with substantial insight for their action research endeavour. They were also useful for Smallsoft's managers in several ways:

- The network models provided images of the communication and knowledge sharing about SPI which the managers trusted as they had been involved in their validation.
- The models contained angles, pointers and clues that the managers had never thought about before. The SPI manager in particular genuinely found the models interesting as a kind of mirror in which he could now see his own organisation in a new light.
- The models had been useful in illustrating and explaining the findings from the researchers to the managers. They proved valuable as a starting

DOI: 10.1002/spip

point for the discussion of an appropriate strategy for SPI in the company, as they emphasized two major problems in the current situation that the managers could agree upon.
- The models were used as a basis for taking decisions about SPI and how to improve the underlying networks that had the potential of pushing the SPI effort forward. These decisions led to actions involving management more strongly and also to the creation of improvement teams across the departments.

It is evident from the SmallSoft case that communication and knowledge sharing in SPI is an integral part of SPI. Researchers and managers should acknowledge this and more attention should be paid to communication and knowledge sharing in SPI efforts. The research literature of small software companies has mostly been concerned with measuring process maturity and the problems that small companies have with maturity models and the CMMI in particular. There is research addressing the need for a closer look at knowledge management from a social perspective (see Kautz 2000; Kautz and Thaysen 2001); there is research addressing knowledge as a commodity to be stored in an experience base (see Conradi and Dingsøyr 2000; Rus and Lindvall 2002); there are reported experiences from building knowledge networks in software organisations (Kautz and Hansen 2008). There are however no reports where the underlying informal networks have been analysed as we have done here.

The SmallSoft action research study also shows an inherently difficult dilemma. On the one hand, the managers want to exercise leadership in SPI, and, on the other hand, there are underlying communities-of-practice in software development (Mathiassen 1998; Wenger and Snyder 2000), which cannot be designed or managed directly. The social network models show some of these communities-of-practice and the managers are with these in hand very aware that they cannot change the social networks as they change formal structures, responsibilities, and tasks. Perhaps, the way we have here used social network analysis points to a way in which the managers can nevertheless navigate and manoeuvre in this landscape.

Our study illustrates that modelling social networks is particularly relevant for understanding SPI activity in small companies. Small software companies are less likely to favour a formal, centralised SPI approach and SmallSoft is no exception here. It is thus reasonable to discuss how the lessons learned in our action research study concerning communication, knowledge sharing and social network analysis may be generalised.

Modelling social networks fits well with a low budget approach to SPI. It is cost-effective to analyse the underlying social networks that are an important part of the infrastructure for a more informal SPI approach in small companies as it has been performed in this study. Small companies lack the economical inclination to invest in a formal, rational, centralised infrastructure.

Modelling social networks enables small companies to discuss, to exploit the possibilities that already exist, and to focus on necessary improvements as a basis for SPI. We thus propose that the way we have modelled social networks can be well transferred to other, similar organisations. On the basis of the described experience, we suggest that it will work for small companies.

We can only speculate about whether it will also be feasible for large software organisations. It is likely that the visualisations from the tool we used will be less useful with more than a hundred developers and other software packages for social network analysis might be more useful.

However, irrespective of the size of company, our study shows that communication about SPI is also necessary in large organisations. Knowledge sharing happens in emergent communities-of-practice that can in part be uncovered with social network analysis. What we know so far is that in order to facilitate discussions that bring improvements forward, the network models must show the networks in a visual way that can be grasped by the involved actors without them being experts in social network analysis.

A CMM-driven strategy can be supplemented with social network analysis and that particular way of looking at the informal organisation. In a CMM-driven strategy the focus is on processes and much less on people (Aaen 2003). A social network analysis thus offers the opportunity to focus simultaneously on how people communicate and how this communication supports knowledge sharing and as such becomes a prerequisite for the organisational change.

The data collection methods and the analysis, and the use of models in discussions and reflections are not specific for SmallSoft. They are all transferable

DOI: 10.1002/spip

to other small software organisations. Thus, we claim generality for the applicability of modelling of social networks and for the performing of social network analyses. What cannot be transferred to other organisations are the specific models, the analyses of SmallSoft, and the specific outcomes of the discussions.

6. CONCLUSION

In this article we have reported on an action research study in a small software company. Data were collected through a case study and a questionnaire that was designed specifically to get information about communication patterns in the social networks involving software developers and their managers. The social network analysis was subsequently utilised in a management reflection to further change and leadership of SPI. Our findings can be summarised as follows:

1. Communication and knowledge sharing about software process improvement follow other patterns than official and bureaucratic channels. It is important to understand the structure of these informal communication networks as they can promote or hinder a particular improvement effort.
2. Such communication and knowledge sharing networks can be studied through social network analysis. Social network analysis and its accompanying tools and techniques offer several very useful analyses. The managers in the case company appreciated these findings and consequently acted upon them. In particular, they deliberately sought to remedy identified shortcomings in the network structures.
3. Social network analysis is very likely to be useful in other small organisations as data collection, visualisation, and the analysis techniques fit well with the particular challenges faced by small software companies, which want to engage in SPI.

The limitations of this action research study are related to its purpose, which has been to explore the usefulness of social network analysis for software process improvement. This exploratory study is only on the basis of a single action research study and that limits its generalisation. The study shows a high validity due its high response rate and due to the fact that findings were iteratively validated. Although the analyses performed with the software tool are reliable and it is therefore likely that a repetition would reach the same outcomes again, the results are only valid for SmallSoft. Hence we do not claim generality for the analyses.

In a continued effort to make social network analysis more useful for software process improvement we will undertake further research with more software companies and improve the questionnaire, as well as the particular analyses and their interpretations.

ACKNOWLEDGEMENT

We wish to thank the developers and managers of SmallSoft for access to the company. For valuable comments on earlier drafts of this article we thank Karlheinz Kautz and the anonymous reviewers.

REFERENCES

Aaen I. 2003. Software process improvement: Blueprints versus recipes. *IEEE Software* **20**(5): 86–93.

Ahern DM, Clouse A, Turner R. 2001. *CMMI Distilled: An Introduction to Multi-discipline Process Improvement*. Addison Wesley: Reading, MA.

Ahern DM, Clouse A, Turner R. 2003. *CMMI Distilled: A Practical Introduction to Integrated Process Improvement*. Addison-Wesley: Reading, MA.

Baskerville R, Pries-Heje J. 1999. Knowledge capability and maturity in software management. *Data Base for Advances in Information Systems* **30**(2): 26–43.

Borgatti SP, Everett MG. 1992. Notions of position in social network analysis. *Sociological Methodology* **22**: 1–35.

Brodman JG, Johnson DL. 1994. What small businesses and small organizations say about the CMM. *16th International Conference on Software Engineering*, Sorrento, Italy, 331–340.

Cater-Steel AP. 2001. Process improvement in four small software companies. *Proceedings 2001 Australian Software Engineering Conference*. Canberra, IEEE Computer Society, 262–272.

Checkland P. 1991. From framework through experience to learning: The essential nature of action research. In *Information Systems Research: Contemporary Approaches and Emergent Traditions*, Nissen H-E, Klein HK, Hirschheim RA (eds). North-Holland: Amsterdam, 397–403.

DOI: 10.1002/spip

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

Conradi R, Dingsøyr T. 2000. Software experience bases: A consolidated evaluation and status report. *Proceedings of the Second International Conference on Product Focused Software Process Improvement*. Berlin, Springer, 391–406.

Cross R, Parker A. 2004. *The Hidden Power of Social Networks: Understanding How Work Really Gets Done in Organizations*. Harward Business School Press: Boston, MA.

Ehrlich K, Valetto G, Helander M. 2007. Seeing inside: Using social network analysis to understand patterns of collaboration and coordination in global software teams.*Second IEEE International Conference on Global Software Engineering*, 297–298.

Granovetter MS. 1973. The strength of weak ties. *American Journal of Sociology* **78**(6): 1360.

Hansen MT. 1999. The Search-transfer problem: The role of weak ties in sharing knowledge across organization subunits. *Administrative Science Quarterly* **44**(1): 82–85.

Humphrey W. 1989. *Managing the Software Process*. Addison-Wesley Publishing Company: Reading, MA.

Humphrey WS. 1992. Introduction to software process improvement. Technical report, CMU/SEI-92-TR-007. Software Engineering Institute, Carnegie-Mellon University: Pittsburgh, PA.

Humphrey WS. 2002. Three process perspectives: Organizations, teams, and people. *Annals of Software Engineering* **14**(1–4): 39–72.

Iversen J, Nielsen PA, Nørbjerg J. 1999. Situated assessment of problems in software development. *Data Base for Advances in Information Systems* **30**(2): 66–81.

Kautz KH. 1999. Making sense of measurement for small organizations. *IEEE Software* **16**(2): 14–20.

Kautz K. 2000. Software process improvement in very small enterprises: Does it pay off? *Software Process Improvement and Practice* **4**(4): 209–226, 10.1002/(SICI)1099–1670(199812)4:4<209::AIDSPIP105>3.0.CO;2–8.

Kautz K, Larsen EAA. 1997. Diffusion theory and practice: Disseminating quality management and software process improvement innovations. *5th European Conference on Information Systems*, Cork, Ireland, 1–16.

Kautz K, Hansen BH. 2008. Mapping knowledge flows. In *Software Processes & Knowledge: Beyond Conventional Software Process Improvement*, Nielsen PA, Kautz K (eds). Software Innovation Publisher: Aalborg.

Kautz K, Thaysen K. 2001. Knowledge, learning and IT support in a small software company. *Journal of Knowledge Management* **5**(4): 349–357.

Kautz K, Westergaard H, Thaysen K. 2001. Understanding and changing software organisations: An exploration of four perspectives on software process improvement. *Scandinavian Journal of Information Systems* **13**: 31–49.

Kelly DP, Culleton B. 1999. Process improvement for small organizations. *Computer* **32**(10): 41–47.

Kotter JP. 2003. Power, dependence, and effective management. In *Organizational Influence Processes*, Porter LW, Angle HL, Allen RW (eds). M. E. Sharpe: New York, 127–141.

Long Y, Siau K. 2007. Social network structures in open source software development teams. *Journal of Database Management* **18**(2): 25–40.

Mathiassen L. 1998. Reflective systems development. *Scandinavian Journal of Information Systems* **10**: 67–117.

Mathiassen L. 2002. Collaborative practice research. *Information Technology & People* **15**(4): 321–345.

Mathiassen L, Nielsen PA, Pries-Heje J. 2002. Learning Spi in practice. In *Improving Software Organizations: From Principles to Practice*, Mathiassen L, Pries-Heje J, Ngwenyama O (eds). Addison-Wesley: Boston, MA, 3–21.

Mathiassen L, Pourkomeylian P. 2003. Managing knowledge in a software organization. *Journal of Knowledge Management* **7**(2): 63–80.

McFeeley B. 1996. IDEAL: A User's Guide for Software Process Improvement. Technical report, CMU/SEI-96-HB-001. Software Engineering Institute, Carnegie-Mellon University: Pittsburgh, PA.

McKay J, Marshall P. 2001. The dual imperatives of action research. *Information Technology & People* **14**(1): 46–59.

Müller C, Meuthrath B, Baumgraß A. 2008. Analyzing Wiki-based networks to improve knowledge processes in organizations. *Journal of Universal Computer Science* **14**(4): 526–545.

Nielsen PA, Iversen JH, Johansen J, Nielsen LB. 2002. The adolescent effort. In *Improving Software Organizations: From Principles to Practice*, Mathiassen L, Pries-Heje J, Ngwenyama O (eds). Addison-Wesley: Reading, MA.

Nielsen PA, Ngwenyama O. 2002. Organizational influence processes in software process improvement. In *European Conference on Information Systems*, Wrycza S, Kautz K (eds): ECIS Gdansk.

DOI: 10.1002/spip

Pries-Heje J, Pourkomeylian P. 2004. Beyond software process improvement: A case study on change and knowledge management. *EuroSPI Conference*. Trondheim, Norway, 2004.

Rus I, Lindvall M. 2002. Knowledge management in software engineering. *IEEE Software* **19**(3): 26–38.

Scott J. 2000. *Social Network Analysis: A Handbook*. Sage Publications: London.

Tichy NM, Tushman ML, Fombrun C. 1979. Social network analysis for organizations. *Academy of Management Review* **4**(4): 507–519.

Villalon J, Agustin GC, Gilabert TSF, Seco AD, Sanchez LG, Cota MP. 2002. Experiences in the application of software process improvement in SMES. *Software Quality Journal* **10**(3): 261–273.

Ward RP, Fayad ME, Laitinen M. 2001. Software process improvement in the small. *Communications of the ACM* **44**(4): 105–107.

Wasserman S, Faust K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press: Cambridge.

Wenger EC, Snyder WM. 2000. Communities of Organizational Factor. *Harward Business Review* **78**(1): 139–145.

Yang HL, Tang JH. 2004. Team structure and team performance in IS development: a social network perspective. *Information & Management* **41**(3): 335–349.

DOI: 10.1002/spip