SOFTWARE PRACTICE IMPROVEMENT

Pouya Pourkomeylian

Gothenburg Studies in Informatics
Report 22, March 2002, ISSN 1400-741X

Software Practice Improvement

Pouya Pourkomeylian

Doctoral Dissertation

Department of Informatics
Göteborg University
Viktoriagatan 13, Box 620
SE-405 30 Göteborg, Sweden
www.informatics.gu.se

AstraZeneca R&D Mölndal, S-431 83 Mölndal, Sweden
pouya.pourkomeylian@astrazeneca.com
www.astrazeneca.com

To: Dana, Donya, and Golaleh

ABSTRACT

Software Process Improvement (SPI) is a systematic approach used to improve the capabilities of software organisations. One basic idea in SPI is to assess the organisations' current practice and improve their software processes on the basis of the competencies and experiences of the practitioners working in the organisation. Implementing improved software processes in practice is, however, difficult. The new processes must be made available on different organisational levels and approached as frameworks for better software practice.

A major challenge for a unit working with SPI efforts is to create strategies and mechanisms for managing knowledge about software development. Knowledge Management (KM) insights are therefore potentially useful in SPI efforts to facilitate the creation, modification, and sharing of software processes in an organisation. A number of studies have, in fact, argued for and illustrated the usefulness of applying KM to SPI. However, much needs to be done to further explore the practical use of KM in the context of SPI.

This thesis reports on research in which KM was used to reflect upon and inspire SPI efforts in a software organisation over a two and a half-year period. The SPI efforts started with an assessment to establish the current capability of the organisation's software practices. On the basis of the findings of that assessment, improvement efforts were planned and carried out to create new software processes. An implementation strategy was developed and different activities were planned and conducted to implement the new processes in different software projects. Further, two complementary approaches to KM, the codified and the personalised, were used to develop a KM strategy and different facilities to support knowledge sharing and continuous SPI efforts in the organisation.

The study can be classified as research in the field of Information Systems (IS), focusing on the practical use of KM insights in SPI efforts and aiming to find answers to the following questions: 1) How can we make SPI happen in practice? 2) What are the main challenges in SPI from a KM perspective? 3) How can KM insights support SPI practice?

The study illustrates that SPI in practice is about managing software knowledge. The key findings are:

- To understand the main challenges in SPI from a practice point of view and make it happen in practice we need to distinguish between processes and practices and between process knowledge and practical knowledge and study the interaction between these types of knowledge as improvements are made.
- SPI initiatives should emerge through personal growth, through knowledge creation, through knowledge adaptation, and through knowledge transformation on the individual level.
- To address the practical issues of managing knowledge in SPI, the efforts should be organised as a project supported by a KM strategy focusing on the most characteristic features of the organisation in question and on improving practice in a stepwise manner. The strategy should include both codified and personalised approaches and change as the software organisation matures.

Keywords: Software Process Improvement (SPI), Knowledge Management (KM).
Language: English
Number of pages: 142

ACKNOWLEDGEMENTS

Making this journey and reaching my goals over the last three years would not have been possible without the support of many people. I am very grateful to all of them. The research was sponsored by AstraZeneca R&D Mölndal, Sweden, in collaboration with the Viktoria Institute Gothenburg, Sweden, and the Department of Computer Science at the University of Aalborg, Denmark.

The research effort required a high level of co-operation at AstraZeneca, where the research was carried out, between the practitioners and myself and between myself and other SPI experts and people in the company who helped me with relevant advice. I wish to thank the following: Ingegerd Andersson, Jesper Arent, Marie Björkqvist, Laine Christensen, Elof Dimänes, Agneta Ekfelt, Daniel Frick, Dulce Gonçalves, Göran Gudichsen, Daniel Johansson, Karlheinz Kautz, Ulla Lagerman-Persson, Sara Lindahl, Lars Mathiassen, Katarina Nellgård, Hans O. Nilsson, Kerstin Nilsson, Mikael Palmblad, Jan Skoog, Anita Vagelin, Michael Yousif, and all my Ph.D. colleagues at the Viktoria Institute.

I wish especially to thank my manager at AstraZeneca, Anita Vagelin, for her support and encouragement and the very useful discussions we had throughout the project. I wish to thank: my co-authors in two of my papers, Lars Mathiassen, Jörgen Hörnell and Staffan Söderberg, all the anonymous reviewers of the papers that form the basis of this research, Janet Vesterlund for her help with my English, and all the personnel at the Viktoria Institute and the Department of Computer Science in Aalborg, Denmark, for their kindness and help during the time I was there. I would like to thank Bo Dahlbom for his valuable suggestions during the finalisation process of this thesis. A special thanks goes to my supervisor, Lars Mathiassen, for his excellent suggestions and criticisms and for his encouragement. I wish especially to thank my mother for the stable ground she gave me in my life. Finally, doing this work would neither have been possible nor as interesting and fruitful to me as it has been without the love, support and understanding of my family, my wife Golaleh and our daughter Dana. I love you and thank you.

Göteborg, December 2001

Pouya Pourkomeylian

CONTENTS

I IMPROVING SOFTWARE PRACTICES BY MANAGING KNOWLEDGE

Pouya Pourkomeylian

1 RESEARCH AIMS

During the last decade Software Process Improvement (SPI) has been used in the software industry as a systematic approach toward improving the capabilities of software organisations. SPI was originally developed at the Software Engineering Institute (SEI) at Carnegie Mellon University and is based on ideas presented by Humphrey (see Humphrey 1989). SPI offers three sets of ideas for improving practice in software organisations: the *management* of SPI activities, the *approach* taken to guide SPI initiatives, and the *perspective* used to focus attention on the SPI goal(s) (see Aaen *et al*. 2001).

The first step in improving the capabilities of a software organisation is to understand the current status of the software development practice in the organisation (Humphrey 1989). One way to do this is to make an assessment based on a model as a road map. In the past years software organisations have used different appraisal approaches to identify what should be improved. The most popular assessment model is the Capability Maturity Model (CMM), which is a stepwise approach to SPI developed by the SEI (Paulk *et al*. 1993). Other approaches include BOOTSTRAP (Kuvaja 1994), SPICE (Thomson and Mayhew 1997), ami (ami 1992), TickIT (TickIT 1995), and TRILLIUM (Thomson and Mayhew 1997). Common to all these approaches is that they apply Total Quality Management (TQM) principles to SPI.

Following an assessment, further improvement activities should be planned and carried out to create new or improved software processes. A process can be defined as a set of tasks and procedures that when performed or executed attain a specific goal (Olson *et al.* 1989). Practice can be defined as the time during which a process is put into action. Once the new or improved software processes are created, the next challenge for those making SPI happen is to implement them in the organisation. Different reports have pointed out difficulties in conducting SPI projects in practice (Goldenson and Herbsleb 1995, Debou 1997). Success with SPI seems to depend on a complex mix of highly interrelated factors acting in different phases during an SPI project. Factors such as scaling the SPI initiative, setting realistic goals, the complexity of organisational changes, and the organisational culture have made it difficult to achieve success in SPI initiatives (Goldenson and Herbsleb 1995, Herbsleb *et al*. 1997, Mashiko and Basili 1997, Johansen and Mathiassen 1998). Still SPI has also been shown to help organisations gain organisational benefits (Humphret *et al.* 1991, Wohlwend and Rosenbaum 1994, Hayes and Zubrow 1995, Larsen and Kautz 1997).

An organisation's software development practices are based on the knowledge and competencies of its practitioners and managers (Arent and Nørbjerg 2000). Mathiassen *et al.* (2001) argue that SPI efforts depend on the implicit, individual knowledge of practitioners in an organisation. To change software development practices, the organisation should improve the practitioners' existing knowledge (both theoretical and practical) of its software practices. Knowledge about the new processes should thus be made available on different organisational levels. However, until recently, according to Kautz and Nielsen (2001) there has been little theoretical and practical understanding of how SPI knowledge can be transferred to all organisational levels.

One major challenge for a unit working with SPI efforts is hence to create strategies and mechanisms for managing knowledge about software development. Other studies have argued for and illustrated the usefulness of applying Knowledge Management (KM) to SPI. Arent and Nørbjerg (2000) analysed how the organisational knowledge creation process can support SPI initiatives. They point out the need for more complex models of knowledge creation and expansion processes than provided by Nonaka and Takeuchi (1995). Baskerville and Pries-Heje (1999) investigated and suggest how KM is used as underlying theory to develop a set of key process areas as a supplement to the CMM in small or medium sized companies. Kautz and Thaysen (2001) studied how knowledge, learning and IT support occur in small software organisations. They argue that IT should not only be used to gather, store, and distribute information. According to them IT should also support learning. Mathiassen *et al.* (2001) studied SPI from a knowledge creation perspective. They argue that the general idea of SPI is to make knowledge explicit and to share knowledge on different organisational levels. Kautz and Nielsen (2001) studied the role of knowledge transfer in SPI implementation and developed a practical framework that can help change agents understand SPI implementation as knowledge transfer. Arent *et al.* (2001) studied SPI as an organisational learning process using Nonaka and Takeuchi's (1995) concept. They suggested two main strategies for learning: 1) the exploration strategy in which the key learning processes are learning by sharing and learning by doing. These processes focus more on changing practice (creating tacit knowledge) than on documenting practice (creating explicit knowledge), and 2) the exploitation strategy in which the focus is on learning by reflection and integration aiming to create explicit knowledge across the organisation in the form of new standards processes and guidelines. These studies indicate that the issues related to knowledge creation, modification and sharing have important roles in SPI initiatives, but to understand where and how to use KM insights to improve SPI practice we need to experiment with and further develop different KM insights in SPI practice. This study has been an attempt in this direction.

SPI efforts are defined as activities carried out to: 1) understand the current status of the organisation's software practices, 2) develop a vision of the desired practices, 3) establish a list of required actions in order of their priority, 4) produce a plan to accomplish the required actions, 5) commit the resources and conduct the plan, and 6) start at step one again (Humphrey 1989). KM activities in SPI are defined as activities focused on the creation, modification, and sharing of knowledge about software development practice and process.

The thesis reports on research in which KM was systematically used to reflect upon and inspire the SPI efforts in a software organisation in a multinational pharmaceutical company, AstraZeneca, during a two and a half-year period. The SPI efforts started with a CMM-based assessment (in May 1999) focused on level 2 Key Process Areas (KPAs) to establish the current capability of the organisation's software practices. On the basis of the findings from the assessment, improvement efforts were planned and made to create new software processes (for software validation, change and version control, and software documentation). Further, 27 templates were developed to be used for documenting the results of software projects. An implementation strategy was developed and different activities were planned and conducted to implement the new processes in different software projects. Further, two complementary approaches to KM were used to develop a KM strategy and different facilities to support knowledge sharing and continuous SPI efforts in the organisation. A second CMM-based assessment made in the organisation (in May 2001) showed that the organisation achieved improvements in all level two KPAs (see Appendix for a brief summary of some of the chosen documents).

The study was organised as an SPI project in collaboration with practitioners working in the software organisation (early 1999 – June 2001). An SPI unit was created to support the SPI efforts and develop strategies and facilities for sharing knowledge about the new practices and processes in the organisation. I have been the driving force in the SPI efforts and the head of the SPI unit. Through these activities, I was able to learn the concept of SPI, understand the organisation's software problems and gain first-hand experience of all the SPI efforts carried out at the software organisation. My experience as SPI manager and practitioner has been inspired in this study by conceptual and theoretical material from a number of sources, as illustrated in table 1. These sources have also helped me evaluate my practice and distil key learning points.

Topic	*References*
Software Process Improvement	(Humphrey 1989), (Zahran 1998), (Paulk *et al.* 1993), (Aaen *et al.* 2001), (Arent and Iversen 1996), (Mathiassen *et al.* 2001)
Knowledge Management and Networking	(Nonaka and Takeuchi 1995), (Hanssen *et al.* 1999), (Scarbrough, H. and Swan, J., ed. 1999), (Scarbrough *et al.* 1999), (Seufert *et al.* 1999), (Augier *et al.* 1999)
Research Approaches	(Patton 1990), (Galliers 1992), (Mathiassen 2000), (Dick 1997 a, 1997b, and 1997c), (Galliers and Land 1987)

Table 1. Overview of primary sources of inspiration

My research can be classified as a study in the field of Information Systems (IS) and can basically be seen as a case study based on action research, although some of the research activities were done using other approaches such as field experiments and practice studies. The goal of the research is to support both academia and practice, thus adopting two purposes: 1) to add to the body of knowledge about how KM is used and can be used to support SPI efforts in practice, 2) to provide benefits to further SPI efforts at AstraZeneca, where this study was conducted. This has been dealt with by exploring answers to the following questions:

1. **How can we make SPI happen in practice?**
2. **What are the main challenges in SPI from a KM perspective?**
3. **How can KM insights support SPI practices?**

I chose Software Practice Improvement as the title of this thesis to move the focus from improving processes to practical issues of improving software practices. Following the sequence of the research questions I first describe the concept of SPI in order to present the major ideas and challenges in SPI. Second, the KM insights chosen to be used in this study are described and discussed. I further present our experiences from using KM insights in SPI in the AstraZeneca case. Different experiences from different phases of our SPI effort are reported. Finally, I discuss each research question in the light of our findings in the AstraZeneca case, and offer some practical advice for how to conduct SPI in practice as a KM effort.

The thesis is divided into three main parts. Part I: Software Practice Improvement, includes six chapters. Chapter one describes the research aims, Chapter two gives the underlying theoretical frameworks for the thesis, Chapter three describes the underlying research approaches, and Chapter four reports on the case in which this research was conducted. Chapter five is a survey of the five papers that form the basis for this thesis, and Chapter six presents the results and offers discussions based on the material presented in the previous chapters. Part II: Research Contributions, includes the five papers that have been published within this project. Part III: Industrial Contributions, presents a brief description of the project’s industrial contributions.

2 THEORETICAL FRAMEWORKS

The central concept underlying this research is SPI. Different approaches originating from insights on KM were used in the study to support the SPI efforts. In this section I give a brief description of SPI ideas (see Aaen *et al.* 2001, Mathiassen *et al.* 2001). I present the CMM as a road map for improving the capability of a software organisation's maturity and the IDEAL model as a framework for organising, planning, and carrying out SPI efforts. I further present the area of KM, including one theory of organisational knowledge creation (Nonaka and Takeuchi 1995) and two complementary approaches for knowledge sharing, i.e. the codified and the personalised approaches (Hanssen *et al.* 1999, Swan *et al.* 1999).

2.1 SPI Ideas

To present the underlying concept of SPI I choose Aaen *et al's.* (2001) MAP because it offers a survey of the ideas within SPI that offer answers to specific practical concerns. According to those authors, SPI addresses three fundamental concerns: the *management* of SPI activities, the *approach* taken to guide the SPI initiatives, and the *perspective* used to focus attention on the SPI goal(s).

Managing SPI: *Organising* SPI efforts is one of the key elements that influence SPI management. It is important that someone within the organisation is assigned responsibility and given the necessary resources for the SPI efforts. An SPI organisation should be dedicated and adapted to the whole organisation. One way to establish a dynamic SPI organisation for improving software practices is to staff the SPI organisation with part-time participants, i.e. as a project. Another way would be to centralise the SPI efforts in a separate group or method department. Experience has shown that it is better to decentralise SPI efforts than to have one centralised SPI organisation that encompasses the whole organisation (Aaen *et al.* 2001, Mathiassen *et al.* 2001). There are several risks in having a centralised SPI group: 1) the group can become a bureaucratic method department, 2) the group might produce solutions that practitioners do not accept, 3) the group runs the risk of becoming alienated and having its results met with indifference in other parts of the organisation. On the other hand, according to Aaen *et al.* (2001), organising SPI efforts as dedicated, localised efforts offers many benefits. One is that it allows SPI experts to collaborate closely with practitioners to define the work procedures that fit their concerns. Another benefit is that resource adaptation can be taken into account. When the SPI effort is established as a localised project, the allocation of appropriate resources will be an integral part of the organisation. But organising SPI efforts as projects includes some risks as well. On the basis of our experience one risk can be that the SPI activities will be confounded in the scope of the project's definition. Another risk can be that the project manager and the project members can be too focused on the deliverables, schedules, and deadlines that they do not have the necessary time for reflection (see Pourkomeylian 2001b).

The second key element influencing SPI management is *planning*. An action plan should be developed to stand as the basis for defining and starting an SPI project, or a number of projects, to implement changes. Many improvement programs have failed as the simple result of no action being taken after appraisals have been made (Paulk 1996). Aaen *et al.* (2001) mention several advantages of creating a plan for the SPI effort: 1) it creates a common understanding about the goals, schedules, and results, 2) it makes it easier to break the SPI project down into a sequence of limited tasks that have specific operational targets, 3) it supports prioritisation and co-ordination, 4) it helps management and SPI project members to form and meet specific commitments, and 5) it serves as a mechanism for communicating progress to ensure proper visibility and understanding of the SPI project. However, several things can go wrong even when there is a plan. Plans can be uncoordinated, meaning that they are not adjusted to other ongoing organisational concerns. As a result, key practitioners can be unavailable when they are needed. Another risk can be that a plan can kill practitioners' motivation and commitment. Planning had impacts on our SPI efforts both as supportive tool and as a risk factor creating feelings of stress (see Pourkomeylian 2001b).

The third key element that affects SPI management is getting *feedback* on improved or new software processes. There are different approaches to measuring SPI efforts (Zahran 1998). Experience has shown that it is not always easy to measure and collect data from SPI efforts. According to Johansen and Mathiassen (1998), measurement activities should be treated as a project in itself. One alternative way to measure effects of SPI efforts is to try to achieve abstract goals, such as go for level 3 or 4 in the CMM model or rely on people's perception of the effects. Among the opportunities that collecting appropriate feedback may provide in the SPI effort is giving legitimacy to the effort and the resources spent by pointing to the positive outcomes in qualitative terms. Another opportunity may be that measurements can serve as an important contribution to creating organisational commitment and motivation. One risk related to measurement according to Aaen *et al.* (2001) is that it may be difficult to argue for the measurements' validity. Another problem is ensuring verifiability in order to establish that measurements are trustworthy, accurate, and reliable. In our case we did not defined how to measure feedback during the improvement phase of our project. We used the direct feedback of the practitioners working with the improvement efforts (see Pourkomeylian 2001b). Later on during the implementation phase of the project we defined different ways of measuring feedback (see Pourkomeylian 2001c).

Approaches to SPI: this aspect of SPI addresses how to design SPI efforts. One key element is that SPI is an *evolutionary* process in which improvements are implemented incrementally over time rather than in a few dramatic transformations. The incremental improvements are continuous, concerted, and cumulative: they follow cycles of assessing, experimenting, and rolling out, and they are oriented toward the individual, project, and organisational levels (see Humphrey 1989, Aaen *et al.* 2001, Mathiassen *et al.* 2001,). An evolutionary approach to SPI has several benefits. One of the important success factors in the SPI approach is the active involvement of the practitioners in identifying, designing, and implementing improvements. Experience-based learning through the SPI effort's life span is another important success factor. Finally, the evolutionary approach allows keeping and leveraging the best elements of existing processes (see Paulk 1996, Aaen *et al.* 2001, Mathiassen *et al.* 2001). One risk related to the evolutionary approach is the slowness of the improvements. People working with SPI might even find themselves in a situation in which they cannot measure any effect because of measurement uncertainties. This was one of the challenges in our case, especially during the improvement phase of our project (see Pourkomeylian 2001b).

Another key element related to approaches to SPI is the use of best practice models of software processes as *norms* or benchmarks for assessing the capability of the software organisation. Such models are also used to formulate a strategy for bridging the gap between norm and practices. Examples of such models are CMM, Bootstrap and SPICE (see Paulk *et al.* 1993, Kuvaja 1994, Thomson and Mayhew 1997). These models can support organisations in comparing experiences and achievements in process improvement efforts. Norms can also provide criteria for setting priorities for improvement areas and implementing stepwise improvements that focus on a limited number of areas at a time. SPI models, according to Aaen *et al.* (2001), have been criticised for focusing too narrowly on SPI objectives and not enough on business goals. One risk in using norms as models can be that the organisations will follow norms for the sake of recognition rather than because they actually need to improve their processes. In our case we used a modified CMM assessment to establish the capability of our software practices. For further improvement efforts we relied on our practitioners' ideas and experiences to create new software processes (see Pourkomeylian 2001a, 2001b).

The final key element related to approaches to SPI is *commitment*, which is the explicit or implicit agreement among participants to strive for shared goals and results. Senior management must actively support the change initiative with leadership, resources, and strategic management (see Aaen *et al.* 2001, Mathiassen *et al.* 2001). If the whole organisation shows commitment to SPI, individuals will be motivated to share experiences, try out new processes, and work together to reach goals. An alternative approach is to base the improvement activities on power. Paulk (1996) argues that a top-down approach instructing everyone to follow the new processes is seldom successful. Another possibility is improvement through individual initiatives. Aaen *et al.* (2001) argue that this might lead to improvements in individual competencies, without sponsorship and co-ordination, and that such initiatives easily lead to islands of excellence rather than improved organisational capabilities. Aaen *et al.* (2001) argue that commitment can go too far. People can become so dedicated to solving problems that they lose sight of the original goals. This can further lead to a loss of perspective on the organisation's long-term improvement visions. People involved in our SPI effort were highly committed to the project. They were all practitioners working with software engineering practices and were interested in solving our software practice problems. But sometimes we needed to remind each other about the scope of our project and try to focus on the goals of our efforts (see Pourkomeylian 2001b).

Perspectives of SPI: the focus of SPI is on software engineering practice (see Humphrey 1989). Aaen *et al.* (2001) summarise the view in the literature on this practice from three basic perspectives. The first perspective of SPI is that it is focused on *software processes*. According to Humphrey (1989) software processes are the set of tools, methods, and practices an organisation uses to produce software products. SPI's total process perspective is holistic, covering all software engineering practices and based on the idea that improved processes can lead to better products and projects (see Zahran 1998, Humphrey 1989, Mathiassen *et al.* 2001, Aaen *et al.* 2001). The benefit of this view is that technology and people are understood in their practical organisational context. Of course the holistic approach adds complexity to SPI activities. The involvement of the organisation's best practitioners in the SPI effort is thus a critical success factor. But there is a conflict between their SPI responsibilities and their software development responsibilities. Another risk comes in underestimating the "people" element and adopting an instrumental view of practitioners. This can result in a lack of participation and further commitment (see Aaen *et al.* 2001, Mathiassen *et al.* 2001). On the basis of my experience as quality manager, having a holistic view on software processes can expand the scope of the areas to be improved and this requires more detailed planning and more resources for conducting the improvement efforts.

Another key element related to perspectives on SPI is people's *competencies*. A successful SPI effort requires competency development in relation to the newly created software processes. The goal of developing competencies is to empower people to expertly use, modify, and adopt the software processes in their software projects in a way suited to their needs. Aaen *et al.* (2001) argue that competence building can create strong commitment to projects and processes. Because people understand and appreciate the process, they are empowered to use their discretion in adapting the processes to meet the needs of both the situation and their customers. One possible risk mentioned by Aaen *et al.* (2001*)* is that the protection of an individual's or group's interests obstructs organisational or project goals. In our case the competence development activities included transformation of knowledge about software development to different organisational levels and teaching practitioners the new processes and coaching them to adapt the new processes to fit their needs. Of course this was a time-demanding process (see Pourkomeylian 2001a, 2001b, 2001c).

The third key element in SPI perspectives is the *context* of software engineering activities. This context provides a ground for improvement efforts on a general level, as well as for customisations for specific needs. The context provides an environment for making it clear who does what, why, when, and how. The context according to Aaen *et al.* (2001) is where individual and organisational competencies merge. Through training, documented guidelines, the repertoire of methods and tools, and other kinds of support, the software process is stabilised into a whole that allows for both the adaptation of existing practices and the adoption of new processes to suit the situation. Such a context helps organisations identify best practices and ease the introduction of new employees into it. It also supports systematic reuse and the development of a professional software engineering culture (see Aaen *et al.* 2001, Mathiassen *et al.* 2001). One risk related to development of the context is according to Aaen *et al.* (2001) that an improvement effort does not effect practice, i.e. the practices remain unchanged. Reuse can lead to carrying out activities because they are part of tradition rather than because they are needed. There is also a risk that the software processes can be defined such that competencies are externalised in procedures and discretionary behaviour is not permitted. This has led to the criticism that SPI causes organisations to become rigid and bureaucratic. However, documented SPI experiences indicate that such criticisms are misplaced in all but a few organisations (Herbsleb *et al.* 1997). In our case we created an SPI unit responsible to facilitate the implementation of new processes in each software project to improve the practice (see Pourkomeylian 2001c).

To sum up, the management of SPI initiatives is based on three ideas: 1) the SPI activities are *organised* in a dynamic fashion, 2) all improvement efforts are carefully *planned,* and 3) *feedback* on effects of software engineering practices is ensured. The approach to SPI initiatives is guided by three additional ideas: 1) SPI is *evolutionary* in nature, 2) SPI is based on idealised, *normative* models of software engineering, and 3) SPI is based on a careful creation and development of *commitments* between the actors involved. Finally, the perspective in SPI is dominated by three ideas: 1) SPI is focused on *software processes*, 2) the practitioners' *competencies* are seen as the key resource, and 3) SPI aims to change the *context* of the software operation to create sustainable support for the actors involved. The holistic view underlying SPI is to focus on software processes as social institutions with a complex interplay of people, methods, tools, and products (Aaen *et al.* 2001, Mathiassen *et al.* 2001).

2.2 SPI Models

A popular model in the field of SPI that is suitable for assisting in organising, planning, and collecting feedback for implementing organisational changes is the IDEAL model (McFeeley 1996). We used the IDEAL model in AstraZeneca's case to organise, plan, and carry out our SPI efforts. We chose to use IDEAL because it is a widely recognised and used model, it is easy to understand and use, it has a strong focus on organisational learning, and it suits our practices.

The most popular and widely used model of the mature software organisation is the CMM, which is a framework that sets out the key factors in an effective software process and describes an evolutionary improvement path from an ad hoc, immature process to a mature, disciplined process (see Paulk *et al.* 1993). The CMM has received some criticism (Bach 1994), but we used it because it was widely recognised and used in SPI efforts, it can be supplemented with other ideas, it allows for flexibility, and it suited our needs.

2.2.1 IDEAL

As is shown in figure 1, the IDEAL model considers five phases (Initiating, Diagnosing, Establishing, Acting, and Learning) of an SPI initiative that provide a continuous loop through the steps necessary for software process improvement.

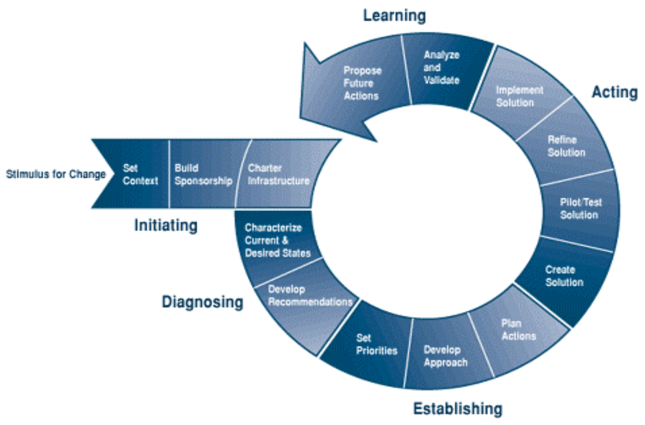


Figure 1. The IDEAL Model (McFeeley 1996)

The intention of the IDEAL model is to present a single picture that is easy to use and helps people remember what to do to establish and conduct successful improvement activities and keep in mind the infrastructure needed to turn the results of an assessment into action. Once the first cycle of SPI has been completed, there is a need to regularly repeat the whole process.

The initial improvement infrastructure of the SPI effort is established in the *initiating* phase. The roles and responsibilities in the SPI effort are defined, and the initial resources for conducting the SPI project are assigned. An SPI plan is created in this phase to guide the SPI project through the completion of the Initiating, Diagnosing, and Establishing phases. In the *diagnosing* phase, appraisal activities are planned and carried out on the basis of a model as a road map to establish a baseline for the organisation's current software problems. In the next phase (*establishing*), strategies are developed for pursuing the improvement suggestions identified in the diagnosing phase. An SPI action plan draft is completed in accordance with the organisation's vision and lessons learned from past improvement efforts. Some measurable goals are also developed and included in the final version of the SPI action plan. Metrics necessary to monitor progress are defined. Commitments are made for resources and training for the people working with the improvement efforts. In the *acting* phase, the suggestions for improvement gathered during the diagnosing phase are planned and carried out in the organisation. Plans are developed to execute pilot projects to test and evaluate the software process created. The objective of the *learning* phase is to make the next pass through the IDEAL model more effective. By this time, the solutions have been developed and documented, the lessons have been learned and documented, the metrics on performance have been collected, and goals have been achieved (McFeeley 1996).

2.2.2 CMM

The CMM describes the process capability of software organisations in five levels. The higher the process maturity, the lower the risk, and the higher productivity and quality become. The idea is that an organisation at a higher maturity level will perform better than one at a low maturity level. The model provides a roadmap for moving from an ad hoc process culture to a culture of process discipline in which there is continuous process improvement. Each maturity level in the CMM is composed of several KPA and each KPA is organised into five sections called common features. The common features specify the key practices that, when collectively addressed, accomplish the goals of the key process area. Figure 2 illustrates the CMM's five maturity levels.

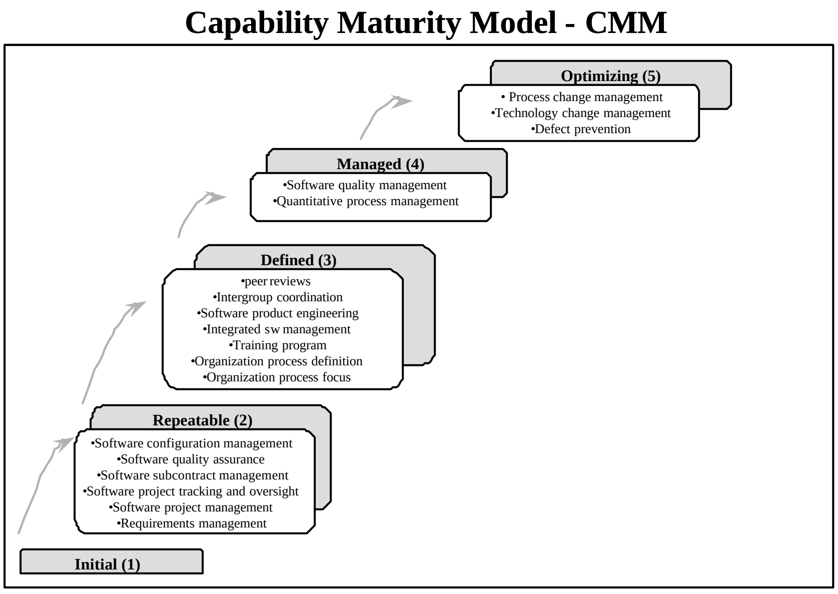

Capability Maturity Model - CMM
Optimizing (5)
• Process change management
•Technology change management
•Defect prevention
Managed (4)
•Software quality management
•Quantitative process management
Defined (3)
•peer reviews
•Intergroup coordination
•Software product engineering
•Integrated sw management
•Training program
•Organization process definition
•Organization process focus
Repeatable (2)
•Software configuration management
•Software quality assurance
•Software subcontract management
•Software project tracking and oversight
•Software project management
•Requirements management
Initial (1)

Figure 2. The CMM model and the five maturity levels (Paulk *et al.* 1993)

The first level is the *initial* level, which has no requirements. This level is consequently often described as an ad hoc level. The software process capability of a level 1 organisation is unpredictable because the software process is constantly being changed or modified as work progresses. Schedules, budgets, functionality, and product quality are generally unpredictable. Organisations that are on this level have no established, basic project management processes, and success in software projects depends entirely on the project team and personal initiatives. There are few stable software processes, and performance does not depend on the organisational features. There are no KPAs on this level. After conducting our first CMM-based assessment we realised that our organisation had some of the characteristic features of being a level one organisation (see Pourkomeylian 2001a).

The second level is the *repeatable* level. Here, the focus is on getting software project management under control to track costs, schedules, and functionality. The necessary process discipline should also be in place to repeat earlier successes in projects with similar applications and to avoid past failures. The software process capability of level 2 organisations can be characterised as disciplined because the planning and tracking of the software project is stable and earlier successes can be repeated. Our CMM-based assessment was focused on level 2 KPAs, although we did not exactly focused on improving those KPAs. Neither did we focus on reaching level two. We selected our own focus areas based on the organisation's requirements and practitioners' ideas (see Pourkomeylian 2001a, 2001b).

Level 3 is the *defined* level, and the focus is on standardising the processes used in the software projects across the entire organisation. This requires the creation of defined processes as a basis for consistent implementation and better understanding. All projects use an approved, tailored version of the organisation's standard software process for developing and maintaining software. The software process capability of level 3 organisations can be characterised as standard and consistent because both software engineering and management activities are stable and repeatable. At level 3 management has good insight into the technical progress of all projects. Costs, schedules, and functionality are under control, and software quality is tracked. In our project we created a few software processes to be used as frameworks for better practice (see Pourkomeylian 2001a, 2001b, 2001c).

The most significant quality improvements, according to the CMM, can begin when the organisation at level 4 (the *managed* level) has initiated comprehensive quantitative measurements to ensure that the software processes operate within statistically predictable limits. At this level, the software process capability of the organisation can be characterised as predictable because the process is measured and operates within measurable limits. Detailed measures of the software process and product quality are collected. Both the products and software processes are quantitatively understood and controlled. On level 3 the software products are expected to be of a predictably high quality.

At level 5 (*optimised* level), the entire organisation is focused on continued process improvement based on quantitative data from level 4. The software process capability of level 5 organisations can be characterised as being in a state of continuous improvement. Improvement occurs both by incremental advancements in the existing process and by innovations using new technologies and methods. At this level, the organisation is typically characterised by a focus on continuous process improvement.

2.3 Theory of Knowledge Management

Dahlbom and Mathiassen (1993) approach knowledge from two different views: 1) a positivistic perspective in which knowledge is like information that can be collected and processed as objective artefacts that can be measured, bought, classified and stored in people, books, or computers, 2) a hermeneutic perspective in which knowledge is not an artefact, but something a person can have. My understanding of what knowledge is agrees with Nonaka and Takeuchi's (1995) definition. They consider knowledge to be preceded by information and, in this process, interpretation, reflection, and action take place. Using Dahlbom and Mathiassen's terminology, knowledge is basically hermeneutic but should be supported by partial positivistic elements.

My understanding of management efforts in KM relies on the creation, modification, and sharing of knowledge through different organisational levels, i.e. individual, group, and organisation (see Shariq 1999, Scarbrough and Swan 1999, Zack 1999). On the basis of this understanding, and because of the nature of our SPI case at AstraZeneca, I chose two KM approaches to support the SPI initiatives: 1) the theory of organisational knowledge creation to support the creation of knowledge and the modification of knowledge to fit it for use in different projects (see Nonaka and Takeuchi 1995) and 2) the approaches of codification and personalisation of knowledge to support the sharing of knowledge on different organisational levels (see Hanssen *et al.* 1999).

Nonaka and Takeuchi's concept (1995) has received some criticism (see Tsoukas 1996, Cook and Brown 1999), but I chose to use the concept for several reasons: 1) it is a well-known approach, 2) it has already been used in SPI with some success (see Arent and Nørbjerg 2000, Arent *et al.* 2001), 3) it deals with different types of knowledge (tacit and explicit) and activities that relate to the creation of knowledge on different organisational levels. The theory of organisational knowledge creation (Nonaka and Takeuchi 1995) was used early in the SPI effort to help us understand the nature of SPI from the perspective of organisational knowledge creation. The first step in our SPI effort focused on creating knowledge about the current maturity of the organisation's software practices. The SPI efforts were further focused to create new software processes based on practitioners' experiences and knowledge about the software practice. With Nonaka and Takeuchi's concept we identified which type of knowledge was created by whom, where, when, and how during the improvement activities. Using this concept helped us to understand the knowledge creation process in our SPI effort and design facilities to support customisation of new processes for individual software projects (see Pourkomeylian 2001a, 2001c).

Nonaka and Takeuchi's (1995) approach is based on the distinction between explicit and tacit knowledge. Explicit knowledge is knowledge that is transmittable in formal, systematic languages. It can be articulated in formal languages, including grammatical statements, mathematical expressions, specifications, manuals, and so forth. It can be transmitted across individuals formally and easily. Tacit knowledge is personal and context-specific, and is therefore difficult to formalise and communicate. It is personal knowledge that is embedded in individual experience and that involves intangible factors such as personal belief, perspective, and value system. Tacit knowledge is difficult to communicate and share in the organisation and must thus be converted into words or numbers that anyone can understand. This distinction is important, especially in software organisations where the tacit knowledge of the professionals must interact closely with various forms of explicit knowledge. The very idea in software engineering is to explicate knowledge in the form of programs to be executed on computers. Software developers spend great effort developing programs, specifications, and models, while at the same time participating in close people-to-people interactions as members of software teams.

According to Nonaka and Takeuchi (1995) organisational knowledge is created during the time the "*conversion*" takes place, i.e. from tacit to explicit and back again into tacit. Knowledge conversion is a "social" process between individuals and is not confined to one individual. Assuming that knowledge is created through the interaction between tacit and explicit knowledge, four different modes of knowledge conversion are possible (Figure 3).

1. From tacit knowledge to tacit knowledge (socialisation that creates sympathised knowledge). The socialisation mode usually starts with building a "field" of interaction. This field facilitates the sharing of members' experiences and mental models. Socialisation involves the sharing of tacit knowledge between individuals. In our case this happened in SPI-workshops, SPI meetings, and customisation meetings when we discussed issues to find solutions for software problems (see Pourkomeylian 2001a, 2001c, Mathiassen and Pourkomeylian 2001).
2. From tacit knowledge to explicit knowledge (externalisation that creates conceptual knowledge). The externalisation mode is triggered by meaningful "dialogue or collective reflection," in which using an appropriate metaphor or analogy helps team members to articulate hidden tacit knowledge that is otherwise difficult to communicate. Externalisation requires the expression of tacit knowledge and its translation into comprehensible forms that can be understood by others. In our case externalisation activities were focused on illustration of the new processes in a way that was easy to understand and easy to share (see Pourkomeylian 2001a, 2001c, Mathiassen and Pourkomeylian 2001).
3. From explicit knowledge to explicit knowledge (the combination that creates systematic knowledge). The combination mode is triggered by "networking" newly created knowledge and existing knowledge from other groups in the organisation, thereby crystallising them into a new product or service. In our case the combination activities were focused on making the new

processes more complete on the basis of practical use (see Pourkomeylian 2001a, 2001c, Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001).

4. From explicit knowledge to tacit knowledge (internalisation that creates operational knowledge). "Learning by doing" triggers internalisation. The internalisation of newly created knowledge is the conversion of explicit knowledge into the organisation's tacit knowledge. In practice, internalisation relies on two dimensions. First, explicit knowledge must be embodied in action and practice. Second, there is a process of embodying the explicit knowledge by using simulations or experiments to trigger learning by doing processes. To implement the new processes in practice we encouraged practitioners to use them as frameworks for better practice. We offered them different services such as training and coaching to teach them how to adopt the new processes into their software projects (see Pourkomeylian 2001a, 2001c).

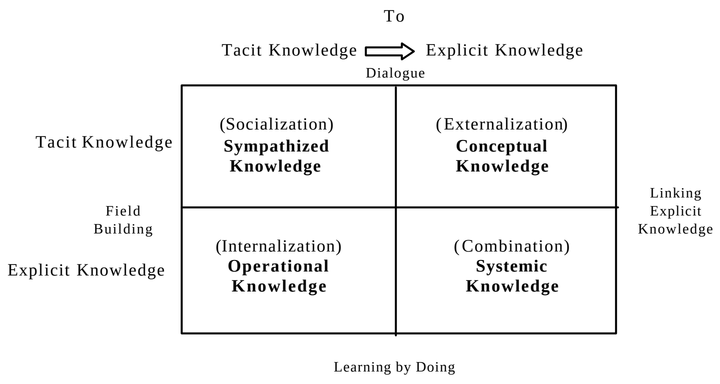

To
Tacit Knowledge
Explicit Knowledge
Dialogue
Tacit Knowledge
Field Building
Explicit Knowledge
(Socialization) Sympathized Knowledge
(Externalization) Conceptual Knowledge
(Internalization) Operational Knowledge
(Combination) Systemic Knowledge
Linking Explicit Knowledge
Learning by Doing

Figure 3. Knowledge created via the four modes (Nonaka and Takeuchi 1995)

As our SPI efforts continued, we needed a further KM approach that could help us create facilities for sharing knowledge and experiences about the new processes and practices. We used the concept of codification and personalisation of knowledge described by Hanssen *et al.* (1999) to create strategies and mechanisms for knowledge sharing through IT-based facilities for sharing codified knowledge and through networks for sharing personalised knowledge. I chose this concept because: 1) it is a concept that is easy to understand, 2) it addresses practical issues, 3) it applies easily to practice, 4) it is suited to AstraZeneca's case (as a pharmaceutical company the quality rules should be addressed explicitly in Standard Operation Procedures (SOPs)), and 5) codification is one key element of SPI.

Two main strategies have been in focus in KM theory and practice: 1) the cognitive model and 2) the community model presenting two different approaches to managing knowledge in organisations (Swan *et al.* 1999). The cognitive model relies on the codification of knowledge objects that are stored in databases from which they can easily be accessed and used by anyone in the organisation (Swan *et al.* 1999). This is what Hansen *et al.* (1999) call the codification strategy and it corresponds to the positivistic view of knowledge according to Dahlbom and Mathiassen's (1993) definition. In contrast, the community model provides a perspective in which knowledge is closely tied to the person who creates it and is mainly shared through direct person-to-person contacts (Swan *et al.* 1999). This is called the personalisation strategy (Hansen *et al.* 1999) and it corresponds to Dahlbom and Mathiassen's hermeneutic view of knowledge.

The focus in the codification or the cognitive model is on elaborating ways to codify valuable knowledge from individuals and to store and reuse it through the use of IT. Knowledge is extracted from the person who originally created it, is made independent of the person, and is reused for various purposes. The computer is seen as a storage facility that can be shared between individuals, and knowledge is codified and later used through "people-to-electronic documents". This approach allows many people to search for and retrieve codified knowledge without having to contact the person who originally developed it. Here, IT and the so-called knowledge agents who are responsible for codifying and storing documents are seen as critical success factors (Hansen *et al.* 1999, Swan *et al.* 1999, Swan *et al.* 2000). This approach to knowledge is contradictory to Kautz and Thaysen's (2001) definition of knowledge and use of IT. They see knowledge as more than an objective codifiable commodity. They argue that knowledge might be individual but that it is socially constructed. They further argue that IT should not only be used to gather, store, and distribute information. It should support learning.

The personalisation or community approach sees the KM issues as socially constructed through interaction between individuals in which collaboration and dialogue, not codified knowledge in databases, are in focus. Knowledge that has not been codified, and probably could not be, is transferred through collaboration, in brain-storming sessions and through person-to-person conversations in networks. A network is seen as a medium for knowledge sharing, both on a personal and on an organisational level (see Augier and Vendelo 1999, Swan *et al.* 1999). IT can play a role in this approach, even though it is not seen as a critical factor for success. The computer is not viewed simply as a storage facility but, more importantly, as a medium for interaction and collaboration. According to Hansen *et al.* (1999), an organisation's KM strategy should rely on the organisation's competitive strategy. They argue that an organisation should take an 80%-20% strategy, i.e. they should choose a primary and a secondary strategy.

In summary, knowledge can be approached both from a positivist perspective, in which knowledge can be collected stored and transferred, and from a hermeneutic perspective, in which knowledge is something that one has and not something one receives. Knowledge is in this thesis preceded by information and, in this process, interpretation, reflection, and action take place. Knowledge is basically viewed from a hermeneutic perspective supported by partial positivistic elements. Management efforts in KM rely on the creation, modification, and sharing of knowledge through different organisational levels (individual, group, and organisation). To support an understanding of the creation and modification of knowledge in SPI we used a KM approach (Nonaka and Takeuchi 1995) focused on the interaction between explicit (referring to knowledge that is transmittable in formal, systematic languages) and tacit (knowledge that is personal, context-specific, and difficult to formalise and communicate) knowledge. According to this approach, knowledge is created during the time that "conversion" takes place, i.e. from tacit to explicit and back again to tacit. To support the sharing of knowledge in different organisational levels we used another KM approach (Hansen *et al.* 1999), which is focused on two main strategies: 1) the cognitive model (the codification approach), which relies on the codification of knowledge objects that are stored in databases and shared through IT-based solutions, and 2) the community model (the personalisation approach) in which knowledge is closely tied to the person who creates it and is mainly shared through networking.

3 RESEARCH PROCESS

Because of the practice-oriented nature of this study and my involvement in both practice and research I basically used case study based on action research as the main approach, although some of the research activities were done using other approaches such as field experiments and practice studies.

To put this research into context as a contribution to IS, I start this section with a brief presentation of approaches to IS research (see Galliers and Land 1987, Galliers 1992) and explain my choice of research approach for this study. I further present SPI as a particular form of research and explain how I used SPI in practice for planning, organising, conducting, and documenting research activities using action research, field experiments, and practice study approaches (e.g. surveys and interviews). I further discuss the strengths and weaknesses of each approach. Finally, I present the data collection process followed in this study and discuss some difficulties in doing this research.

3.1 Approaches to IS Research

According to Galliers and Land (1987), IS is a meta-subject that spans many disciplines in the social sciences, in business, and, sometimes, in the natural sciences. IS, as Galliers and Land (1987) define it, is an applied discipline and not a pure science. Thus the measure of the success of research is whether our knowledge has been improved to the extent that it can be applied in practice. They further argue that, if the results of IS research fail to be applicable in the real world, then our endeavours are irrelevant.

Many researchers have discussed different research approaches to the field of IS (see McFarlan 1984, Mumford *et al.* 1986, Galliers 1992, Mathiassen 2000). Galliers and Land (1987) distinguish between two tendencies in IS research: 1) the traditional, empirical, positivistic research approaches such as laboratory experiments and surveys, which are inspired by the natural sciences, and 2) the interpretative research approaches such as field experiments, qualitative case studies, and action research.

Traditional empirical research originally dominated IS research. As much as 85 percent of published IS research undertaken by leading U.S. institutions used to be of the traditional kind (Vogel and Wetherbe 1984). The field of IS has grown and become much broader as concerns IS and its relations to the organisation and the people it serves (Land 1986). Galliers and Land (1987) consider that both IS academicians and practitioners have begun to realise that it is appropriate to extend the focus of IS research to include behavioural and organisational considerations. They consider further that this wider view brings with it added complexity, greater imprecision, the possibility of different interpretations of the same phenomena, and the need to take these issues into account when considering an appropriate research approach. The problems inherent in IS research that arise from this view of the subject and that call for new approaches are described in McFarlan (1984).

Galliers and Land (1987) point out that the scientific paradigm is not the only, nor indeed always the most appropriate, basis for IS research. Greater thought regarding the choice of research method is required, as is a wider interpretation of what is seen as acceptable research. In choosing a research approach for the field of IS, Galliers and Land (1987) advise that one first should consider the nature of the information systems and then look at what one hopes to gain from undertaking research in the area.

More recently, Mingers and Stowell (1997) and Mingers (2001) have argued that IS is more than simply the development of computer-based solutions. IT is now so fundamental within society that IS as a discipline must concern itself with the general evolution of human communication. Thus, it has to draw upon a wide range of disciplines that encompass very different research traditions. This puts IS in a position similar to other management areas such as organisational studies, which are also characterised by a plurality of research paradigms, each with particular research methods. Mingers (2001) argues that research results will be richer and more reliable if different research methods, preferably from different (existing) paradigms, are routinely combined.

The focus of our study was to improve software practices at AstraZenaca through SPI research using KM insights. We organised, planned, and conducted activities in two very closely interrelated ways to reach our objectives: 1) research practice and 2) industrial practice (see figure 4). Following Galliers and Land (1987) and Mingers (2001) the research approach used in this study is therefore basically interpretative and based on action research in combination with field experiments and qualitative practice studies based on surveys. This position will be further elaborated in the following.

As illustrated in figure 4, this research relates industrial practice and research practice within SPI. Three main areas directly support (full arrows) our SPI practice at AstraZeneca: 1) SPI frameworks, 2) KM frameworks, 3) SPI experiences in other organisations. The results of the research contribute directly to SPI frameworks and indirectly (dashed arrow) to KM frameworks in the form of new SPI and KM knowledge. The main goal was to support the industrial practice of SPI by gathering state-of-the-art insights from IS, SPI, and KM to improve software practice at AstraZeneca. To do this, I visited SEI in Pittsburgh to learn more about the CMM and the IDEAL model. I attended an international SPI conference to share my experiences from our SPI project with other practitioners and researchers and to learn from the experiences of other companies doing SPI. I followed different industrial courses and seminars (e.g. software validation, configuration management, and project management) to better understand the concept of software engineering. I read reports of both successes and failures in SPI in other organisations in order to understand the opportunities and challenges in SPI in practice. I also studied the research approaches used in other SPI projects, which helped me in my selection of an approach for this study. The SPI insights helped me to organise, plan, and conduct our SPI project to improve practices at AstraZeneca.

I studied different KM theories (Nonaka and Takeuchi 1995, Hanssen *et al.* 1999, Swan *et al.* 1999) and read different KM reports done in the field of IS and SPI (Baskerville and Pries-Heje 1999, Stelzer *et al.* 1998, Halloran 1999, Arent *et al.* 2001, Arent and Nørbjerg 2000, Kautz and Nielsen 2001, Kautz and Thaysen 2001). I attended two international KM conferences (at which I presented two papers) to share our practical experiences in using KM in our SPI project with other researchers and to learn from others' experiences in applying KM in IS. Using KM insights in this study helped us develop an adapted KM strategy including two complementary approaches to KM and develop mechanisms for facilitating knowledge sharing in the SPI unit at AstraZeneca.

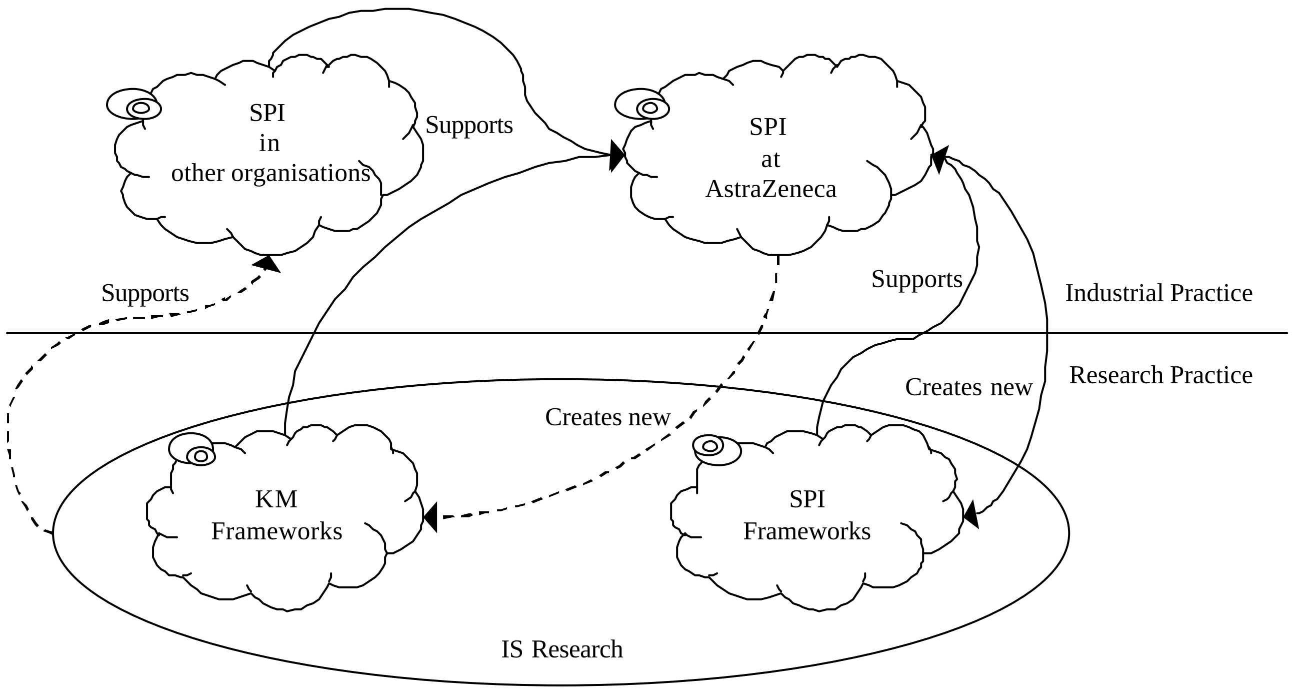


Figure 4. Relation between research and practice in the AstraZeneca case

The results of a problem analysis done in early 1999 in one of AstraZeneca's largest software development groups in Mölndal, Sweden (Development IS, DevIS), showed a need for improving software practices and providing guidelines for understanding the SOPs addressing different authorities' quality requirements. The director of DevIS initiated an improvement project called Software Process Improvement at DevIS (SPID, headed by me as project manager). The purpose was to understand the existing problems, improve the organisation's software processes and practices, and provide guidance to better understand the authorities' quality rules. At that time, I worked at DevIS as quality manager responsible for issues related to IS/IT quality at AstraZeneca. Owing to the nature of our improvement project and my involvement in both industrial practice and research practice, I chose an action research approach as the main approach, which helped me to integrate research and practice. We could introduce change at the same time the research was going on, and we could involve other practitioners in our project. However, as the study proceeded and the focus of our activities changed to other issues, such as creating new processes and testing them in software projects or studying and evaluating current networking facilities at AstraZeneca, we understood that it was necessary for us to use other approaches as well. We thus combined our main research approach (action research) with other approaches - field experiments and surveys - to be able to address other activities.

The research approach, which explains the use of action research in combination with field experiments and practice studies, is collaborative practice research, which is presented and discussed by Mathiassen (2000).

3.2 Collaborative Practice Research

Mathiassen (2000) describes the main concern in collaborative practice research as establishing well functioning relations between research and practice. Improving practices is the distinguishing feature of collaborative practice research and action research in general (Baskerville *et al.* 1996, Mathiassen 2000). Ideally, a collaborative researcher focuses on keeping the research process tightly connected to practice to get first-hand information and in-depth insight. At the same time, the researcher must structure and manage the research process in ways that produce rigorous and publishable results. To achieve these goals, the collaborative research approach is based on action research in combination with practice studies and field experiments. It hence follows the recommendations suggested by Mingers (2001) and Mingers and Stowell (1997).

3.2.1 Overall design

To structure the research process in the organisation as an action research effort I used the IDEAL model (see McFeeley 1996). The model can be used to guide the development of a long-range, integrated plan for initiating and managing an SPI project. I organised, planned, and conducted different SPI activities in order to implement new software processes and KM strategies and mechanisms in the whole organisation.

I used practice studies to focus and structure the research process and findings. A modified CMM-based assessment (see Appendix, the CMM questionnaire) was used twice: 1) first to gather data on the current capability of the software practices at DevIS, and 2) second to evaluate the impact of the use of the new software processes in software projects. The findings from the first assessments were integrated with findings from other quality efforts made in the organisation, and a software process improvement proposal report was created (see Appendix, the software process improvement report 1999) as the basis for decision making among senior management in the area of improvement activities. Another assessment (see Appendix, the SPID Questionnaire) was made (May 2001) before ending the research project in the organisation to evaluate the effects of the SPI unit's services to practitioners and software projects. The results of the second assessment and the SPI unit assessment were integrated in a final report submitted to the steering committee of the project (see Appendix, the final project report for SPID 2001). Finally, an interview-based analysis was made to evaluate the organisation's current knowledge sharing practices and suggest strategies and mechanisms for further activities for sharing knowledge in the SPI unit.

The research has also used a qualitative field experiment approach to focus on specific issues, such as creating and verifying new software processes (see Appendix, the software guidelines), developing IT-based solutions for knowledge sharing (see Appendix, the Electronic Process Library (EPL)), and developing and testing templates for documenting the results of software projects (see Appendix, templates). Using this approach we developed and tested new software processes in the organisation. We illustrated the new processes both as documents (guidelines) and process maps. To support the documentation process in software projects we developed 27 templates and made them available through the EPL. To support knowledge sharing in the organisation we developed and implemented KM strategies and mechanisms. To support the implementation of the new processes in all software projects we developed and offered new services such as training, customisation meetings, and feedback measurement.

The approaches used in this SPI research and practice are listed in table 2.

SPI Research	SPI Practice
Action Research	• Conducting an SPI project in two and a half years using the IDEAL Model • Implementing new processes in the whole organisation

	• Implementing strategies and mechanisms for managing knowledge in the SPI unit
Practice Studies	• Evaluating the current maturity level of the organisation, the CMM Assessment I • Analysing the current knowledge sharing practices in the organisation • Self-assessment, i.e. assessing the results of the SPI effort, the CMM assessment II • SPI unit assessment
Field Experiments	• Creating three new software guidelines • Creating 27 templates for documenting software projects • Creating process maps • Developing a process library and illustrating the process maps in an IT-based system (the EPL) • Developing KM strategies and mechanisms • Developing new services for the SPI unit for facilitating the implementation of processes in the whole organisation

Table 2. Approaches in research and practice

3.2.2 Action research

Dick (1997a, p 2) defines action research as follows:

> *"Action research is a process by which change and understanding can be pursued at the one time. It is usually described as cyclic, with action and critical reflection taking place in turn. The reflection is used to review the previous action and plan the next one".*

Action research is a research paradigm through which the researcher is able to develop knowledge or understanding as a part of practice. The most characteristic features of action research are:

1. *Cyclic*, i.e. similar steps tend to recur in similar sequences during the life span of the research. A typical action research cycle includes *planning, acting, observing,* and *reflecting* (and then planning again). The later cycles are used to challenge and refine the results achieved in the earlier cycles. A typical cycle is as shown in figure 5.

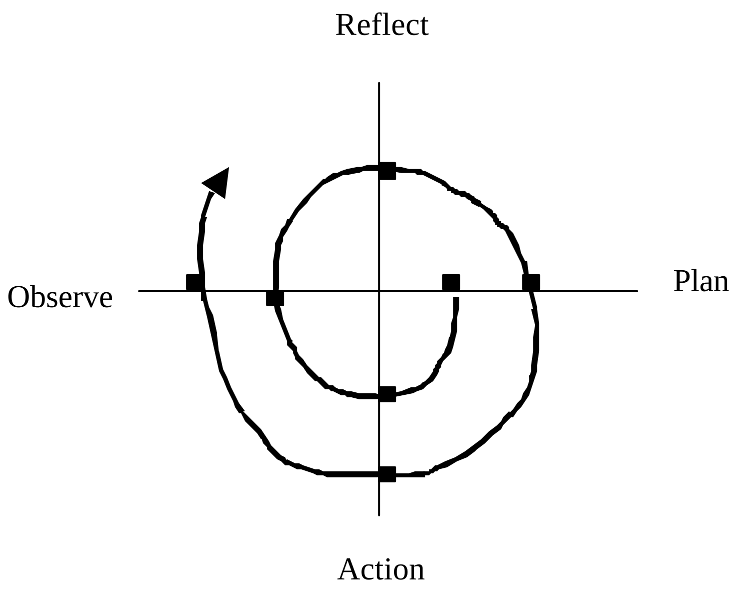


Figure 5. A typical cycle in action research

In our study we followed the cyclic process (plan, act, observe, and reflect) during the whole SPI project. We developed an SPI plan, which addressed different steps needed to create each software process. For each software process we then repeated the cyclic steps (planned improvement efforts, performed SPI workshops, observed and analysed our actions, and critically reflected on the results of our efforts, which gave input to our next steps). To implement new processes in the software projects we developed an implementation strategy including actions, schedules, deliverables, roles, and responsibilities in order to structure the implementation efforts. Stepwise actions were then taken to implement the new processes in each project through cyclic efforts. Our implementation activities consisted of offering different services such as training, advisory/coaching, and customisation meetings through which we customised the new processes for each software project (see Pourkomeylian 2001c).

2. *Reflective,* i.e. the researcher regularly and systematically critiques what he/she is doing in the field study. Critical reflection on the research process and the outcomes are important elements of each reflective cycle. Reflection was an important factor in our project. Reflection on the research process was formally structured through regular meetings both with my supervisor and my manager at AstraZeneca. We reviewed plans and the outcomes of our actions and refined plans for the next actions (if needed). We also encouraged reflection-in-action (see Schön 1987), especially during the implementation of new processes in software projects. We encouraged practitioners to critically use the new processes in their projects and reflect-in-their-actions while using the processes to try to modify them to fit to their situation. We asked them to evaluate the usefulness of the processes in practice and give us feedback on further improvements (see Pourkomeylian 2001a, 2001c).
3. *Participative*, i.e. the practitioners are active participants and are involved in the research activities as co-researchers. The practitioners and the researcher plan the first or subsequent steps together, which are then carried out. The researcher meets with the practitioners to recollect and critically examine their experiences. On the basis of the same process, they decide what the next steps should be, what information is needed or the desired outcome to be pursued, and what method to use. In our case all the SPI efforts were organised, planned, and performed in direct co-operation with practitioners. The new processes were created on the basis of practitioners' ideas and experiences about software practice. To implement the processes in a

project, I together with the practitioners from the software project through a customisation meeting decided the steps needed for the project to adapt the processes to their conditions. I then followed the project (through follow-up meetings) to see how the customisation activities worked for that project. In the follow-up meetings we discussed (the practitioners and myself) how they followed the steps agreed at the first meeting, what went right, and what did not. Then, on the basis of the outcome of the project, we discussed the next steps and planned for further implementation efforts for the project (see Pourkomeylian 2001c). I as a researcher together with my supervisor organised, planned, and conducted the research activities during the life span of the research. We divided the research activities into five phases as is recommended in the IDEAL model and reported on the project in five published papers presented in different international conferences and journals.

4. *Qualitative*, i.e. most action research is qualitative. Action research deals more often with qualitative data than with quantitative data. Qualitative data are data collected on the basis of people's personal ideas, opinions, feelings, and/or experiences about a specific question or problem area. Qualitative data can be collected in three ways: 1) in-depth, open-ended interviews, 2) direct observation, and 3) written documents (Patton 1990). Qualitative inquiries often deal more with language than with numbers. In our case all the data collected were based on practitioners' personal experiences from software practices at AstraZeneca. We collected their opinions about: current software practices, knowledge sharing through networking facilities at the company, the impact of the new software processes on the software practices, and the impact of the SPI unit's services (such as training, advisory efforts/coaching, and customisation meetings both among practitioners and the software projects) (see Pourkomeylian 2001c).

Good action research uses multiple cycles (Dick 1997c). Within each cycle, the researcher should use multiple data sources and try to disprove the interpretations arising from earlier cycles. Action research is often seen as giving answers specific to particular questions that cannot be generalised to apply to other questions. This means that one is often not able to generalise from action research. Claims can often only be made about the particular situation studied. This may be seen as one of the major disadvantages of action research (Dick 1997b). Another important weakness in the action research approach according to Mathiassen (2000) is the limited support provided to help structure the research process and findings. To see how we addressed these issues see section 3.4.

3.2.3 SPI as action research

Following the IDEAL model a project organisation was designed during the *initiating* phase of the project. The organisation consisted of a steering group (including six software managers) responsible for making decisions about the project and the results and for dedicating resources to the project, a reference group (consisting of five project managers and developers and one SPI consultant) responsible for working with improvement activities for creating new or modified software processes, an SPI working group (consisting of two external consultants and myself) responsible for leading the SPI improvement activities and documenting the results, and finally an SPI expert, responsible for giving advice to the steering committee. I was appointed project manager of the SPI project and my supervisor SPI expert. Several meetings were held with the director of DevIS to discuss SPI and different improvement strategies.

During the *diagnosing* phase that used a practice study approach to establish the current maturity level of the software organisation, a modified CMM assessment based on a method called QBA (Questionnaire Based Assessment (Arent and Iversen 1996)) was made (May 26th 1999) that focused on three different software development projects chosen from two different software development groups. The following KPAs were studied in the assessment: 1) Software Project Planning, 2) Requirements Management, 3) Software Project Tracing and Oversight, 4) Software Quality Assurance, and 5) Software Configuration Management. The data collected were analysed statistically (by an external consultant and myself), and proposals were developed for improving software development projects based on the results of an analysis of the qualitative data collected in the assessment, the software process improvement literature, and other quality improvement findings from earlier quality activities done in the organisation (see Appendix, the software process improvement report 1999).

Further, in the *establishing* phase of the project, a local research group (consisting of the director of the software organisation, my supervisor, and myself) was formed to draw up improvement strategies, plan for research on SPI in the organisation, and establish the necessary resources for the SPI research and practices. The research group met twice a year throughout the two and a half-year period to plan and organise SPI initiatives and the related research and discuss difficulties and problems.

In the *acting* phase, to focus on the creation of software processes and on KM strategies and mechanisms, I used a field experiment approach in combination with practice studies. The field experiment approach provides direct access to practices that are partly controlled by the researcher (Mathiassen 2000). Using this approach I created an implementation plan in direct collaboration with practitioners. Further, the reference group was formed and began to plan and carry out improvement activities over a period of four months, resulting in the creation of new software processes.

The implementation plan was created through a Participatory Implementation Workshop (PIW-workshop) to address the implementation activities necessary to initiate the new processes in the whole organisation (see Andersson and Nilsson 2001). Two external consultants were invited to hold a PIW workshop at the software organisation to help us identify the most important factors needed for creating an implementation strategy for new software processes. The workshop was held in May 2000 at AstraZeneca. One SPI consultant, one project manager, all involved in SPID, the two external PIW consultants, and myself participated in the workshop (see Pourkomeylian 2001c, Appendix, the implementation plan for SPID improvements 2000).

After presenting the new software processes and the institutionalisation model to management, the steering committee accepted the new software processes and decided to implement them throughout the organisation. The implementation activities were scheduled between June 2000 and June 2001. A practice study approach was used to study the current KM practices of the organisation (January – February 2001). Seven already established networks were examined by interviewing members of the networks. On the basis of the results of the interviews, a KM strategy including both codified and personalised approaches and mechanisms was developed to support knowledge sharing efforts in the SPI unit (see Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001).

Finally, in the *learning* phase we evaluated the effects of using the new software processes and the services that the SPI unit offered by making different assessments in May 2001. The first was a CMM-based assessment (studying four software projects using the same CMM-based assessment as was used in 1999). The second assessment (see Appendix, the SPID questionnaire) was a self-developed questionnaire focusing on areas such as: 1) the training sessions in which new software processes were introduced in the organisation, 2) the SPI unit itself (what the practitioners knew about the SPI unit and its services), 3) the customisation meetings in which the SPI unit together with practitioners adapted a process for their software project, 4) the software processes (whether they worked in practice), and 5) the software templates (how usable they were). The results of the second assessment and the evaluation of the SPI unit were analysed and summarised in a report presented to the steering committee on June 18th, 2001. The project delivered the second version of the software processes as planned, based on the practical use of the processes in software projects.

3.3 Data Collection and Analysis

On a more specific level, the data collection strategy adheres to four basic criteria: 1) systematic gathering of data through documenting SPI meetings, 2) data collection through interviews, 3) multiple sources of data (e-mail data collected), and 4) participatory observation. We documented all meetings and collected e-mails, memos, reports, project specifications and project plans (table 3 shows the range of data collection). This multiple source data collection strategy helped us in the systematic gathering and reliable recording and transcription of data, which in turn helped ensure the validity of our empirical findings.

Data source	**Explanation**
Meeting notes	Project meeting notes from SPI meetings
Minutes of Meetings	Project meeting protocols from SPI meetings
Electronic Messages	Improvement suggestions, questions from practitioners
SPI Documents	SPI plans, schedules, improvement report, implementation report, final project report
Company Documents	Standard Operation Procedures (SOPs), Software Guidelines
Maturity Assessments	CMM reports (I&II)
Feedback	Training, customisation meetings, electronic messages
Direct involvement	Leading the improvement and implementation activities over two and a half years
Interview notes and records	Report from the interviews and tape recording
SPI unit Assessment	The SPI unit report

Table 3. List of data sources

The data had different sources: meetings notes from the SPI meetings, project meeting protocols from SPI meetings, e-mails from practitioners giving us feedback on different matters, all project-related documents such as project plans and schedules, all company-related and relevant documents such as SOPs and quality guidelines, the results of the CMM assessments, feedback collected through training and customisation meetings with practitioners, data collected through direct involvement in all the project activities during the three years, interview notes, tape recordings of interviews, and data collected from the SPI unit assessment.

Due to the large amount of data collected in the project, we decided first to obtain an overall picture of the data by source ordering and then to make a rough sorting into categories. On this basis relevant documents were selected for further analysis. The data collected were then analysed in different ways according to their type and the purpose of the analysis.

The data collected in the two CMM assessments and the SPID assessment were analysed by the external consultants and myself. The results of all activities were presented to those responsible in the company (often the steering committee of the project). Having responsible people read everything that related to them formed the validation of the analyses before publication. Comments on the analyses were used to improve them before submission for final publication.

Since the research project was an ongoing longitudinal study, we were not able to say at any point in time that data were complete. We continued to work with gathering data based on the practical use of the software processes to continuously improve them. Deciding when to stop looking at practices and start making analyses and reporting results was a constant dilemma.

3.4 Opportunities and limitations

Having different roles (researcher and project manager) in this research project improved my skills in both. As a practitioner I had the benefit of having access to practice. I was aware of the history and the background of the problem area and I could encourage people to participate in the research project. As I matured in my role as researcher, I learned how to use different tools to approach the problem area, changing focus when needed and systematically studying the problem area, collecting and analysing data and reporting results.

There were, however, some problems related to having two roles. First, as the project manager, I sometimes felt stress to deliver practical results on time (see Pourkomeylian 2001b). Second, as quality manager at the company, I was interested in solving many problems and in expanding the scope of the project, which posed a great challenge with respect to maintaining a useful scope in the project (the steering committee and the reference group of the project helped keep the improvement activities within the defined scope of the project). Third, I had to write and report the results of each activity in two different ways: one to the steering committee of the project and another in an academic context. This was a very time-consuming effort considering my other responsibilities at the company. Last but not least it was difficult to critically approach my own actions, especially having two roles and striving to reach goals in both research and practical areas. What would be the results of the SPI project if I did not have my internal role as quality manager at AstraZeneca or my researcher role? Would the project have any success at all? These are questions that I am not able to answer. But I have consistently sought to limit my own bias by confronting my practice and findings with the research results of others and by iterating my own research findings based on valuable feedback from the reviewers of my papers.

Each of the approaches used in this study had specific strengths and weaknesses. The action research approach was chosen because of its strong integration of research and practice. However it offered limited support for structuring the research processes and the findings (Patton 1990, Galliers 1992, Mathiassen 2000). To address this issue I used the IDEAL model to organise, plan, and conduct all research activities during the life span of the research. Moreover, we decided to test all processes and services created in practice (testing new processes, templates, and implementation services: training and customisation meetings to all projects at DevIS). These field experiments helped me to focus on specific issues and practical results. Finally, by using a practice study approach, we focused on the practice of software engineering and KM. To integrate this research and practice we used the findings of the assessments and the interviews for further activities to achieve other planned targets (e.g. the result of the first assessment was used to establish the current maturity level of DevIS and support decision making and choice of improvement efforts).

The most fundamental problem related to data analysis was, however, the problem of generalisation, in particular because I had only one case to study. Patton (1990) argues that applied research seeks limited generalisations. Applied research findings are typically limited to a certain time, place, and condition. The generalisation issue in applied research concerns the effectiveness of specific interventions on specified cases under specifiable conditions. However, this still permits the researcher to seek patterns that cut across different cases in a number of different places and in a number of different groups. In my study I have learned from other SPI and KM experiences in different organisations (Wohlwend and Rosenbaum 1994, Goldenson and Herbsleb 1995, Hayes and Zubrow 1995, Herbsleb *et al*. 1997, Mashiko and Basili 1997, Larsen and Kautz 1997, Johansen and Mathiassen 1998, Arent and Nørbjerg 2000, Arent *et al*. 2001, Kautz and Nielsen 2001) and provided useful knowledge to improve the software practices at AstraZeneca. I have subsequently used general texts on SPI and CMM research and on KM in general to test the relevance of the findings of this research and to place my own findings in a broader IS research context (see figure 4). As I mentioned earlier, the goal of my research was to support both academia and practice. The practice-oriented results of this research have been in use within DevIS. To use these results in other organisations a certain degree of modification is needed. However, I believe that the academic contribution of my study can together with other research findings from different research projects contribute to the creation of more general knowledge for approaching SPI from a KM perspective (see the "creates new" arrow in figure 4).

4 INDUSTRIAL SUMMARY

The study was done at AstraZeneca R&D Mölndal. AstraZeneca is a research-driven pharmaceutical company with a range of products designed to fight disease in different areas of medical need. The company was formed in April 1999 by the merger of Astra AB and Zeneca Group PLC. AstraZeneca's research base and product portfolio is designed for seven areas of medical need – cancer, cardiovascular, central nervous system, gastrointestinal, infection, pain control and anesthesia, and respiratory. AstraZeneca has more than 50,000 employees worldwide. It has research and development (R&D) centers in Sweden, UK and the USA and R&D headquarters in Södertalje, Sweden. The company has some 10,000 R&D personnel and a US $2 billion R&D investment in 1999, extensive global sales and a marketing network employing over 25,000 people, and 12,000 people employed in production in 20 countries.

4.1 The DevIS

The R&D IS function supports the R&D organisation in AstraZeneca with IS/IT support. The R&D IS (December 2001) is a global organisation divided into four global functions. The global Discovery IS and Global Development IS are responsible for supporting the core business (Discovery and Clinical R&D) with IS support (developing, purchasing, and maintaining software solutions). The Global Technical Computing & Information Services is globally responsible for supporting the whole R&D organisation (both the core business areas and the R&D IS functions) with a technical IS/IT infrastructure. The Global Information Architecture develops and maintains the standards, guidelines, and principles related to seamless information integration capabilities. Each global IS function is represented locally in different sites around the world (see figure 6).

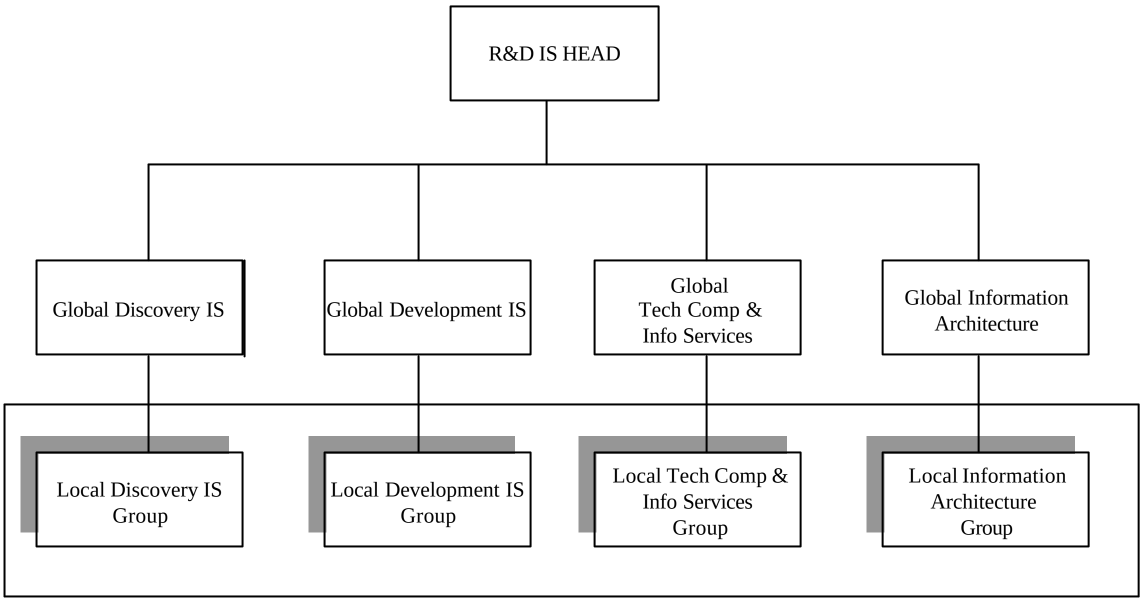

R&D IS HEAD
Global Discovery IS
Global Development IS
Global Tech Comp & Info Services
Global Information Architecture
Local Discovery IS Group
Local Development IS Group
Local Tech Comp & Info Services Group
Local Information Architecture Group

Figure 6. The R&D IS organisation

This research started in early 1999 before the merger between the two companies in an IS organization called Clinical Research and Information Management (CRIM) at the former Astra Hässle in Sweden and continued later in the new IS organisation, which then changed its name to DevIS, which is one of the local Development IS Groups within the Global Development IS function. DevIS supports clinical and pharmaceutical projects, Regulatory Affairs and Product

Strategy and Licenses at AstraZeneca R&D Mölndal. DevIS is also responsible for influencing the development of the global clinical research processes and IS/IT tools at AstraZeneca. DevIS comprises more than 90 people including contractors, most of whom have backgrounds in IS/IT.

Many regulatory authorities require pharmaceutical companies and their software organisations to comply with GXP (Good *Manufacturing* Practice, Good *Clinical* Practice, and Good *Laboratory* Practice) rules. GXP rules are the authorities' quality requirements to pharmaceutical companies for ensuring patient health, the quality of processes (e.g. clinical studies or software development) and the quality of products (e.g. tablets or software). One fundamental requirement is that DevIS must be able to show the authorities, in documented evidence, that software development activities (e.g. software change control, software validation, and data processing and storage) are being performed in compliance with quality requirements. Therefore every software project regulated by GXP requirements should carefully apply all quality rules and be able to show by documented evidence that the software is compliant with the related GXP requirements. The company long ago adopted SOPs that explicitly describe how the company's software quality rules should be addressed in software projects. These SOPs should be applied in all software projects regulated by GXP requirements.

Employees of DevIS are basically engaged with software development, software maintenance, and software operation activities. The software development activities occur in three forms: 1) development of totally new software products (software development), 2) developing or changing existing software products (software maintenance), 3) implementing globally developed or purchased software locally at AstraZeneca Mölndal. A typical software development project at DevIS is scheduled to take between six months and one year and includes analysis, design, construction, testing, and validation. Software maintenance activities can consist of changes in the code or developing a completely new application for existing software products. Software products in DevIS include the software and all related documentation (e.g. user requirement specification, test plan, validation plan, validation report, user manuals etc.).

The Problem Area
In early 1999 the results of an informal problem analysis showed that CRIM's software project practice needed improvement. There was also a need to provide guidelines to understand the SOPs and the GXP rules. Many practitioners working in different software projects pointed to this subject for improvement by sending e-mails to an analysis group responsible for gathering software professionals' ideas for improvement. Management at that time did not have detailed knowledge of the depth of the problem or how to improve the software project practice (see Pourkomeylian 2001a).

At that time we did not know how and where to start improvement efforts. However, I had heard about successful results in other organisations using SPI and the CMM for improving the capabilities of software organisations. After further study of the SPI literature the need of using two approaches for the improvement activities became clearer: 1) a structured and systematic model for planning organising, analysing, and improving software practices, 2) a model for focusing specifically on software project problems. After meetings with the director of DevIS and an SPI expert (my supervisor) to discuss different approaches to improvement activities we decided to start an SPI project, (SPID), using the IDEAL model for planning, organising, and running SPI activities, and using the CMM to focus on level 2 key process areas (see McFeeley 1996, Goldenson and Herbsleb 1995, Hayes and Zubrow 1995).

The first step was to establish SPID's organisation. The SPID organisation, illustrated in figure 7, was established in April 1999.

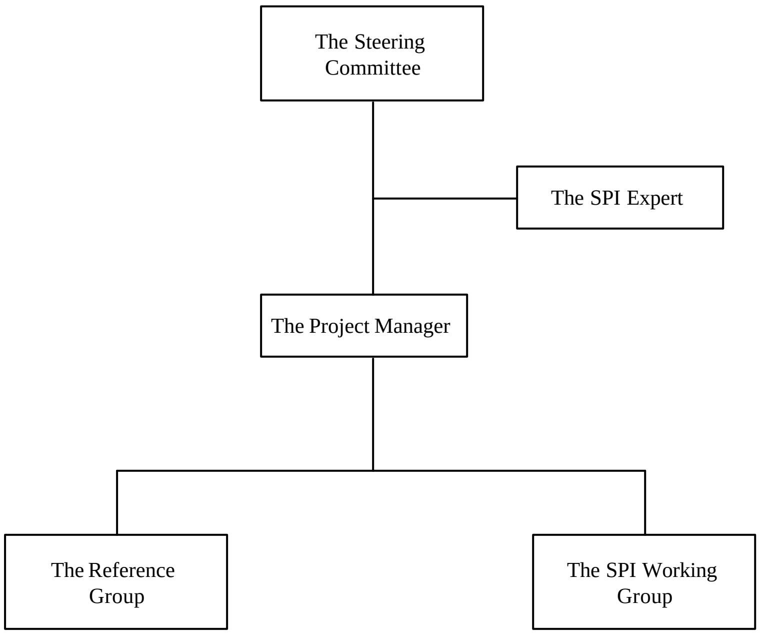


Figure 7. The SPI Organisation

The structure of the organisation later changed when the project finalised the improvement phase and entered the implementation phase (see figure 8).

The second step in SPID was to diagnose the current maturity level of DevIS's software projects through a CMM-based assessment (see Appendix, the questionnaire). On the basis of the results of the assessment, I produced an improvement report in October 1999 addressing six improvement activities, with a specific focus on software documentation and software validation to satisfy one important quality requirement, namely that the pharmaceutical industry must document all software engineering activities to comply with health authority regulations. In the case of inspections the company must be able to show documented evidence that a specific task, e.g. that implementation of change in a software product, has been done in accordance with predefined standard procedures.

The steering committee of SPID decided to give priority to software documentation, software validation, and software change management processes through:

- Creation of a template library including templates for the documentation of software development activities such as: user requirement specification, design specification, test plan, and validation plan.
- Creation of a software documentation process including a minimum documentation level for documenting the results of software projects.
- Defining processes for software validation, software change management, and document version control.

As the next step, an improvement plan was created by me, and the SPI working group started to work on improvement activities, which resulted in the creation of three new guidelines: a software documentation guideline, a software validation guideline, a software change control and document version control guideline - and developed the template library (see Pourkomeylian 2001a). The newly created software guidelines were presented to the steering committee and further modified on the basis of the committee's feedback (see Appendix for a brief description of some of the chosen documents). We further conducted a MAP analysis to better understand the nature of our SPI project and to be able to plan and conduct further SPI activities (see Aaen *et al.* 2001, Pourkomeylian 2001b).

An implementation plan was created and presented to the steering committee. The steering committee of the project accepted the refined software processes and the implementation plan and decided to implement the processes throughout DevIS (see Pourkomeylian 2001c).

The acceptance of the software processes and the decision to implement them throughout the whole organisation was a first important milestone for the initiative. The implementation activities were scheduled to take place between August 2000 and June 2001. One important aim was to change the context in which the new software processes should operate. Therefore, among other activities, a trainee program was scheduled for all practitioners at DevIS. The implementation phase also included further improvement activities in which the processes would be enhanced on the basis of experience of using them in practice. The implementation phase aimed to result in a new version of the software processes in June 2001 (see Pourkomeylian 2001c).

To do this, SPID's organisation changed as shown in figure 8.

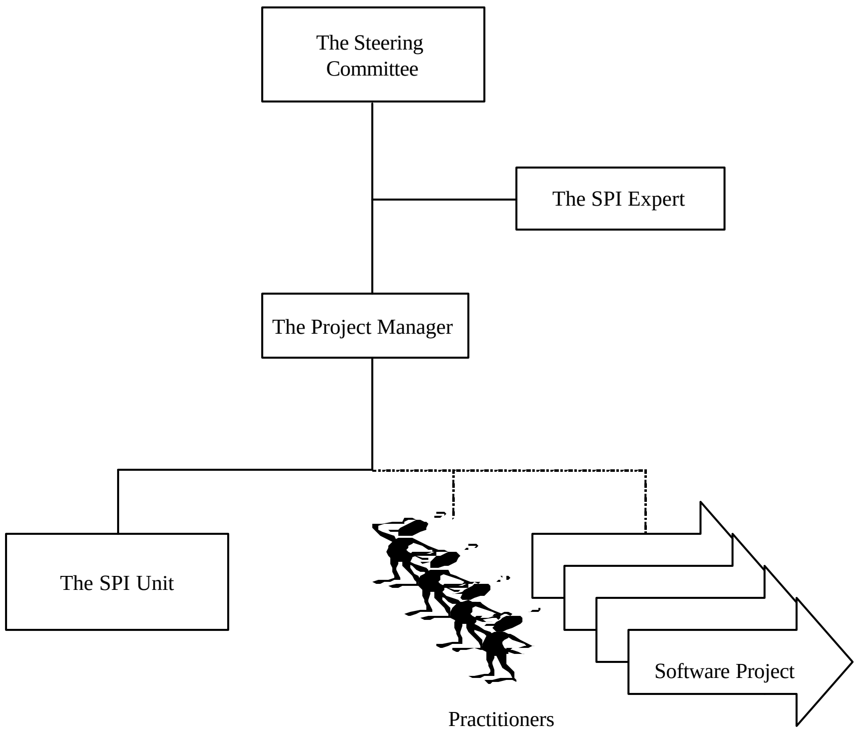


Figure 8. The organisation of SPID (through implementation)

An implementation group including one SPI consultant (the SPI Unit) and myself was responsible for conducting all implementation activities (e.g. training, customisation meetings, advisory, feedback measurement, and project follow-up). The practitioners and the software projects were the main targets for the implementation activities. The practitioners participated in different training sessions to learn the new processes and the members of new software projects started to use the SPI unit's services and implement the new software processes in their projects (see Pourkomeylian 2001c).

4.2 The CMM Assessment I

One focused SPI practice was to make a CMM assessment to establish the current maturity of software practices at DevIS. A modified CMM assessment (see Appendix, the CMM questionnaire) based on QBA (see Arent and Iversen, 1996) was made on May 26th 1999 for three different software development projects chosen from two different software development groups. These projects were chosen on the basis of special criteria:

- Software development or maintenance project which includes development of new applications and have been finished and resulted a system
- Typical software development project at DevIS
- Not a "one-man" project
- Developer working with software development for at least 3 man months.
- Have a clear role as project leader or program manager and developer

The approach was chosen for the assessment part of this study because the CMM's level two KPAs focus on the areas that fit the problem area studied in DevIS. Further, the modified version of QBA helped us to use the terminology used in DevIS and to focus more on the organisation's goals than simply routinely following the CMM questionnaire. The results of the first assessment showed that DevIS at that time was a level one organisation and improvements in all KPAs were needed (see Pourkomeylian 2001a).

4.3 The Knowledge Management Facilities

One great challenge during the implementation phase of the project was to transfer new knowledge about the new software processes to all practitioners in the organisation. To facilitate this we developed a KM strategy based on two complementary approaches, i.e. the codified and the personalised to support knowledge sharing within and outside the SPI unit (see Swan *et al.* 1999, Hansen *et al.* 1999). On the basis of this strategy the SPI unit developed two KM mechanisms to support the creation, modification, and transfer of knowledge within and outside of the SPI unit (see Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001).

4.3.1 The Electronic Process Library (EPL)

To make the SOPs, guidelines, and templates easy to understand and easy to access we decided to illustrate them as process maps. An informal project team including one leader, a developer, two Human Computer Interaction (HCI) experts and two process consultants worked together to create the process maps and develop a process library and illustrate it through the SPI unit's homepage on the company's intranet. Figure 9 illustrates the EPL (see Appendix, the EPL).

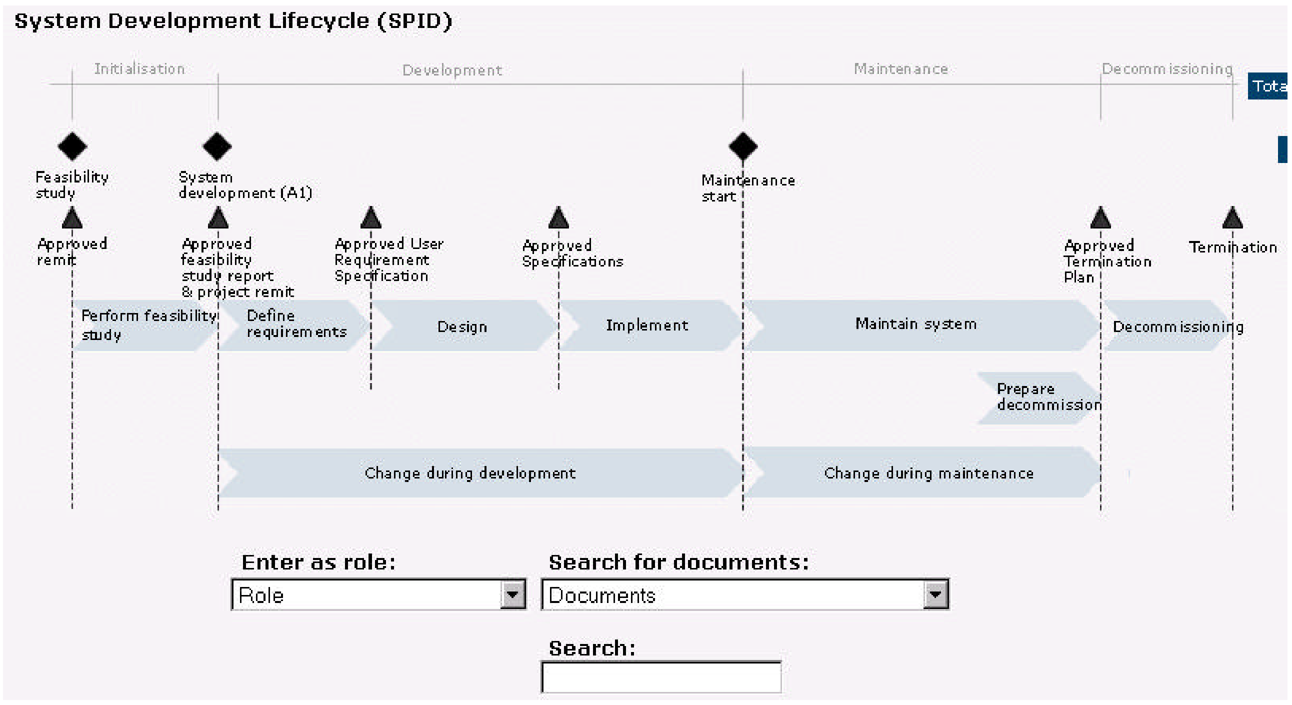


Figure 9. The Electronic Process Library (EPL)

The EPL offers several services to the practitioners. By using the EPL one can easily find knowledge (by knowing his/her role in a software project) about all phases in the System Development Lifecycle, his/her responsibility in each phase, the deliverables for which the person is responsible and the documents that should be produced in each phase. By doing this we illustrated codified knowledge stored in paper documents as process maps and offered facilities to access answers to the "what" questions, i.e. what are my responsibilities as developer in the design phase? What are the deliverables? Which documents shall I produce?

The EPL also offers possibilities for on-line documentation and shows examples of best practices from earlier software projects. It furthermore shows the names of other persons who have experience in different issues (e.g. software validation, test or software change management) from earlier software projects. In this way we tried to facilitate the knowledge transfer between practitioners. Another service of the EPL is a searching mechanism that makes it possible for the practitioners to find useful knowledge about the SOPs and the guidelines. We also developed a Frequently Asked Questions (FAQ) forum through the SPI unit's homepage to make the most common questions and answers about the SOPs, the processes, and the quality rules available to all practitioners (see Mathiassen and Pourkomeylian 2001, Appendix, The EPL).

4.3.2 THE NETWORKING FACILITIES

To be able to solve practical problems in software projects, practitioners need more than answers to "what" questions. We should offer them customised solutions and hands-on support to help solve specific problems. To answer the "how" questions and facilitate the creation and transformation of knowledge related to the software processes, the SPI unit developed additional networking facilities. We participated in different networking forums: 1) between the SPI unit and the software projects: to facilitate the implementation of software processes in each software project the SPI unit arranged "customisation meetings" with each newly started software project, 2) at the organisational level: the SPI unit arranged training sessions for practitioners, managers, system owners and other staff to create knowledge and collect feedback about the new processes, 3) within the SPI unit: to share knowledge and experiences among the members of the SPI unit about experiences from different software projects, new quality rules, and different customised solutions, 4) at the global level: the SPI unit is a part of a global network for quality and compliance at AstraZeneca. In this way the SPI unit participates in networking activities with other quality and compliance groups within AstraZeneca to share knowledge and experiences from SPI activities and new quality and compliance rules (see Pourkomeylian *et al.* 2001).

4.4 The CMM Assessment II

To identify the effects of the SPI activities the second CMM-based assessment was made at DevIS on May 21st 2001 using the same CMM-based questionnaire that was used in assessment I. Four software projects were chosen for the assessment on the basis of special criteria:

- Software development, maintenance or legacy system validation project which has used the newly created software processes (the software guidelines and templates)
- Typical software development project at DevIS
- Not a "one-man" project
- Have a clear role as project leader or program manager and developer
- Have used the services offered by the SPI unit (e.g. training and customisation meetings)

Seven project managers and developers from four projects answered the questionnaire, and the data collected from the assessment were analysed by two SPI consultants and myself. I developed proposals for further improvements and presented them to the steering committee (see Appendix, the final project report for SPID).

Although we did not focus especially on achieving CMM level 2, our improvement activities had effects on the level 2 KPAs (table 3). These assessments should be approached as "improvement indicators", i.e. the improvement percentage is an indicator that some degree of improvement has been achieved according to a qualitative measurement of what practitioners working with the processes have experienced in their software projects.

The Key Process Areas (KPA)	**The CMM I 1999**	**The CMM II 2001**	**Improvement Indicators**
Software Requirement Management	50%	58%	16%
Software Project Planning	53%	57%	8%
Software Project Tracking and Oversight	38%	51%	34%
Software Quality Assurance	46%	59%	28%
Software Configuration Management	30%	57%	90%

Table 4. Improvement indicators

This assessment indicates that, in general, we have achieved improvements in all areas. The most visible improvement was achieved within Configuration Management, Project Tracking and Oversight, and Software Quality Assurance. It is also worth noting that our organisation today has a more homogenous maturity level than it had two years ago.

4.5 The SPI Unit Assessment

To find out whether the services offered by the SPI unit had led to a change of practices in software engineering, the SPI unit made (May 21st 2001) a self-assessment asking the same persons in the four software projects involved in the CMM assessment II about five different areas of the SPI unit's services (see Appendix, the SPID questionnaire): 1) the training program: how it changed the practitioners' understanding about the new processes and the quality rules, 2) customisation meetings: whether this service simplifies the documentation process in their software projects, how the feedback to the SPI unit has worked and whether it has led to changes in the software processes or templates, 3) the new software guidelines (the software documentation guideline, the software validation guideline, and the software change and version control guideline): how they helped practitioners understand the software documentation, software validation, and software change management issues, 4) the templates: how they simplified the documentation process in software projects.

The data from this assessment were collected and analysed by two SPI consultants and myself. The results of the analysis were documented and reported to the steering committee by myself (see Appendix, the final project report for SPID 2001). The results show that the SPI unit's services in general increased practitioners' understanding about the three areas in focus, i.e. software documentation, software validation, and software change management. It has in some cases caused changes in practice as well. This change has had different forms:

1. Increased practitioners' understanding of quality requirements and how to document a software project.
2. Simplified the choice of which documents should be produced in a software project.
3. Established new routines in software projects.
4. Caused a sharper focus on validation issues in software projects.
5. Simplified the validation and test activities.
6. Cultivated a culture to give feedback to the SPI unit about the usability and usefulness of the processes and templates.

One of the most important effects of the implementation efforts has been that these efforts have cultivated a commitment towards continuous improvement among practitioners. 72% of practitioners said that the software project's feedback to the SPI unit has led to changes in templates. This shows that practitioners have been actively involved in improving the templates on the basis of practical use. The results of the analysis also showed that the templates, the Software Change and Version Control, and the training program need further improvement (see Appendix, the final project report for SPID).

4.6 Plans for Further Improvements

On the basis of the findings in CMM-assessment II and the SPI unit assessment I wrote a final report and presented it to the steering committee of the project to finalise the first cycle of the SPID project (June 19th of 2001). I presented the results of the project and reported the project as finalised. My suggestion to the steering committee for further improvement activities focused on the following points:

- Developing and establishing a Requirement Management guideline for addressing among other things tractability issues and review activities.

- Developing and establishing a Project Management Guideline for addressing among other things the project planning and tracking issues including project planning, project review, reuse of project experiences, and change management (focusing on the project plan)
- Improving and establishing the Quality Review efforts.
- Improving the change and version control process and supporting the process with a change and version control tool.
- Improving the training program.

The steering committee of SPID accepted the results of the project, announced the project as finalised (June 19th of 2001), and decided to go on with further SPI activities in the organisation on that basis (see Appendix, the final project report for SPID 2001).

5 RESEARCH SUMMARY

This chapter summarises the five papers including the research approaches used seen from academic viewpoints. Table 4 shows the details of each paper. The full text of each paper and the complete list of co-authors are given in part II.

No	Title	Authors	Publication	Primary approach (Galliers 1992)
1	Knowledge Creation in Improving a Software Organisation	Pourkomeylian	IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations, April 7-10, 2001, Banff, Canada	Action research
2	Analysing an SPI Project with the MAP Framework	Pourkomeylian	SJIS, Scandinavian Journal of Information Systems 2001	Practice study
3	An Approach to Institutionalisation of Software Processes	Pourkomeylian	Tenth International Conference on Information Systems Development, September 5-7, 2001, London, England	Field experiment
4	Managing Knowledge in a Software Organisation	Mathiassen and Pourkomeylian	Submitted to the Journal of Knowledge Management. Earlier version (Knowledge Management in a Software Process Improvement Unit) published in Managing Knowledge Conference April 10-11, 2001, University of Leicester, England	Practice Study
5	Knowledge sharing in a Software Process Improvement Unit	Pourkomeylian, Hörnell, and Söderberg	The Second European Conference on Knowledge Management, November 8-9, 2001, Bled, Slovenian	Practice Study

Table 5. Overview of research papers

5.1 Knowledge Creation in Improving a Software Organisation

Pourkomeylian, P. (2001a). Knowledge Creation in Improving a Software Organisation. *IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

This paper reports the results of the improvement efforts in SPID analysed as an organisational knowledge creation process. The study explains how knowledge is created through transformation between tacit and explicit knowledge and through interaction between different organisational levels of actors.

The academic practice underlying this paper can be characterised as action research as I joined with the practitioners to contribute to the SPI project by planning and making the assessment and leading the improvement activities to create new software processes in the organisation.

The paper discusses on a general level how the knowledge creation process in our SPI project takes place. Who are the actors involved in this process and on what organisational level does this knowledge creation process take place? The paper uses Nonaka and Takeucchi's (1995) theories on organisational knowledge creation. From the perspective of this framework the paper argues that certain types of knowledge in an SPI project deal with the fundamentals of SPI such as: management of SPI initiatives, how such an initiative should be guided, and issues related to SPI's focus on target(s) (see Aaen *et al.* 2001). Other types of knowledge deal with Software Engineering (SE-knowledge) such as: project planning, quality assurance, change control, and configuration management (see Pressman 1997).

The paper concludes that: " two related knowledge domains, SPI-, and SE-knowledge are involved in the knowledge creation process in an SPI project. The project manager of the SPI project, software managers, and assisting consultants are the key actors involved in the creation of SPI-knowledge. The software engineers and the SPI project manager are the key actors involved in the creation of SE-knowledge. The knowledge creation process in the case of SPI-knowledge happens mostly on the individual level and sometimes on the group level. The knowledge creation process in the case of SE-knowledge happens mostly on the group level and sometimes on the organisational level." The main contribution of the paper is an analysis of our SPI effort from an organisational knowledge creation perspective and an illustration of how the knowledge creation process occurred in our SPI project. It further helped us understand the need for having strategies and mechanisms for managing knowledge through the SPI effort's life span.

5.2 Analysing an SPI Project with the Map Framework

Pourkomeylian, P. (2001b). Analysing an SPI Project with the MAP Framework. *Scandinavian Journal of Information Systems*, Vol. 13, pp. 147-165.

The purpose of the second paper was to analyse the SPID project on the basis of the MAP framework (see Aaen *et al.* 2001) in order to better organise, plan, and carry out further SPI efforts. By knowing the nature of our SPI efforts we better understood the risks and opportunities involved and could better manage further SPI efforts.

The research approach underlying the academic practice of this paper can best be characterised as practice study, since the researcher reflects on the project and analyses it on the basis of the MAP framework.

In August 2000 an SPI expert and I analysed the SPI project in the software organisation. For this purpose, we used the MAP framework. The goal was to identify the concrete elements of the SPI effort and to identify crucial SPI features that we had missed or could emphasise further. For each fundamental SPI element, the SPI expert and I determined whether and to what extent the ideas had been applied and followed during the course of the SPI project. We discussed the reasoning behind utilising or not utilising each specific SPI idea and evaluated the effects that pursuing or not pursuing an idea would have on the SPI effort. We also identified actual pitfalls in every SPI idea. I completed the MAP analysis in February 2001 for improvement activities carried out after August 2000. Finally, we assessed the usefulness of the MAP for understanding and implementing SPI initiatives.

Having applied the framework to analyse an initiative in a beginner organisation, it is argued that a novice organisation can succeed in its SPI initiation if it gains and maintains management and practitioner commitment, starts with a focus on a few carefully selected software processes that need improvement, and uses the existing knowledge about the software practices in the organisation. The study led to a number of lessons relevant to future SPI projects in novice organisations and confirms the research underlying the MAP framework.

The paper concludes that: “A first SPI initiative should always be organised as projects with specific goals, deliverables, and resources. A novice organisation should focus on the SPI concept rather than on model-based recommendations like those of the CMM. It is highly recommended that a MAP analysis be made early in the initiation phase of the first SPI initiative to understand the most characteristic features of the SPI project.” The main contribution of the paper was that it helped us focus our SPI efforts on learning about our problems and improving a few processes through small, stepwise learning cycles rather than blindly following CMM with the intent of reaching an abstract level of maturity.

5.3 An Approach to Institutionalisation of Software Processes

Pourkomeylian, P. (2001c). An Approach to Institutionalisation of Software Processes. *Tenth International Conference on Information Systems Development*, London, England.

This paper presents how we worked in a practical way with the institutionalisation of new software processes in DevIS and how we cultivated a change process in a stepwise manner to facilitate the implementation of processes in the organisation.

The research approach underlying this paper can best be characterised as field experiments based on action research, since we created and adopted a change process suitable for DevIS’ conditions and implemented it in practice, measuring feedback to identify weaknesses, problems, and further improvement areas. I was involved in a practical way in the efforts to organise, plan, and carry out the change process to institutionalise the new software processes in the organisation and add benefits to the SPI project.

The institutionalisation approach used in this study was created in a PIW-workshop (Andersson and Nilsson 2001) that included a risk analysis to identify the most important risks and the measures needed to address them. Our implementation strategy was based on a direct introduction of the new software processes in the entire organisation. It consisted of three main elements: 1) a training program for all practitioners in the organisation, 2) process adaptation for all software projects, and 3) feedback measurement and continuous improvements of software processes. The heart of our institutionalisation model was the SPI unit, which was specifically responsible for the development and maintenance of the organisation's software processes. The unit also helped every software project to adapt the new software processes in their project, to interpret the contents of the SOPs, and to follow up the software projects evolved. The continuous improvement activities resulted in the creation of the second version of the software guidelines.

The paper concludes that: "Process institutionalisation should come through personal growth through learning and should start in a small way to first reach process internalisation. Software process implementation should be carried out stepwise through detailed planning and follow-up. Management should encourage practitioners to reflect-in-action for creating variation-in-action, support the creation of an infrastructure to help in the adaptation of the software processes for every software project, and address continuous improvement. New roles should be defined for coaching practitioners and teaching them new software processes on both the group and individual levels." The main contribution of the paper was that it helped us understand that improving the organisation's current software engineering practice is a matter of cultivating change through knowledge sharing and learning.

5.4 Managing Knowledge in a Software Organisation

Mathiassen, L. and Pourkomeylian, P. (2001). Managing Knowledge in a Software Organisation. *Submitted to the Journal of Knowledge Management*. An earlier version: "Knowledge Management in a Software Process Improvement Unit" was published in the *International Conference on Managing Knowledge: Conversations and Critiques, University of Leicester, England.*

This paper explores the practical usage of state-of-the-art insights on KM to support innovation in DevIS. Two complementary strategies for KM, the codified and the personalised, are presented together with practical insights on how to apply these to organisational practice (see Hanssen *et al.* 1999, Swan *et al.* 1999). In this paper, we describe and evaluate current KM practices related to SPI at AstraZeneca using the framework of Hanssen *et al.* (1999) and Swan *et al.* (1999). On the basis of this evaluation we discuss the choice of KM strategy to support further initiatives in the SPI unit. The paper further reviews the conventional wisdom from the SPI literature on how to manage knowledge in software organisations in the light of DevIS' case.

The research approach underlying this paper can best be characterised as practice study. On the basis of an analysis and interpretation of AstraZeneca's case we developed a KM strategy and mechanisms to support the creation and modification of knowledge and knowledge sharing in the SPI unit.

The paper argues that the SPI unit should adopt a KM strategy based on two complementary approaches, i.e. the codified and the personalised to support knowledge sharing within and outside the SPI unit. We conclude that is advisable for SPI efforts to explicitly address their KM strategy. Each software organisation has to find its own balance between personalised and codified approaches. This balance needs to be dynamically adjusted as the organisation matures, and the KM strategy should differentiate between different types of SPI services. We should, however, be cautious when attempting to generalise the specific findings of this study. This is first because SPI initiatives and software organisations are different, but also because the SPI initiative at AstraZeneca has reached a certain level of maturity and the choice of KM strategy is most probably dependent on the maturity level of the software organisation in question.

The paper further presents a review of conventional approaches to SPI (the IDEAL and the CMM) in the light of KM strategies and our specific experiences, thereby providing some concerns for consideration in other SPI initiatives. The IDEAL model (McFeeley 1996) and the CMM (Paulk *et al.* 1993) are expressions of KM strategies. KM is not explicated as part of the models but is implicitly integrated into the ways in which software organisations are advised to professionalise their practices.

The paper concludes that: " A KM strategy should be defined early in the SPI project. A KM strategy for SPI should include both codified and personalised strategies. The KM strategy of an SPI initiative should change as the software organisation matures. This leaves us with the basic conclusion that SPI is knowledge creation and management within software organisations. It is advisable to explicitly address the KM strategy issue, but each organisation must find its own balance between personalised and codified strategies and this balance needs to be dynamically adjusted as the software organisation matures." The main contribution of the paper to our project was that it helped us develop different strategies and facilities to address different KM challenges within our SPI unit.

5.5 Knowledge Sharing in a Software Process Improvement Unit

Pourkomeylian, P., Hörnell, J. and Söderberg, S. (2001). Knowledge Sharing in a Software Process Improvement Unit. *The Second European Conference on Knowledge Management*, Bled, Slovenian.

This paper presents an analysis of current networking facilities in AstraZeneca Mölndal to improve and further develop the SPI unit's knowledge sharing efforts. Further, it presents an improved mechanism for supporting the knowledge sharing activities in the SPI unit based on the results of the analysis of the current networking practice and state-of-the-art advice on networking.

The research approach underlying the academic practice of this paper can best be characterised as practice study. We analysed and evaluated the current networking facilities in DevIS through interviews of practitioners in the organisation and used the results, together with knowledge from state-of-the-art KM, to propose strategies and mechanisms for knowledge sharing through networking.

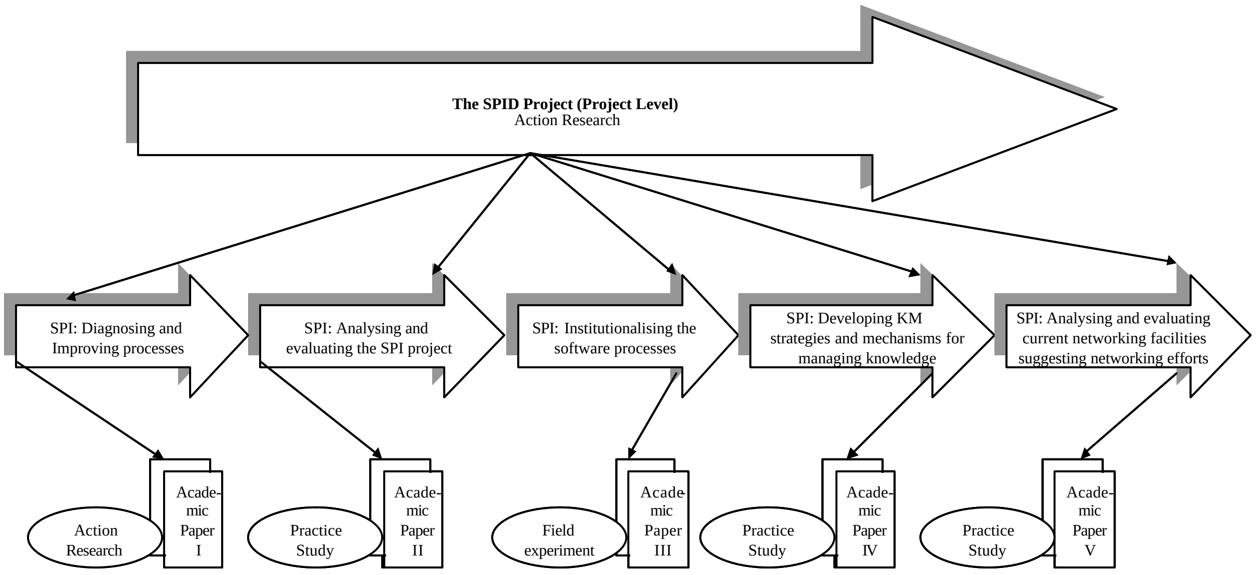
The paper presents a theoretical framework focused on networking based on the works of among others, Hanssen *et al.* 1999, Swan *et al.* 1999, Scarbrough and Swan 1999, Zack 1999, and Seufert *et al.* 1999, to analyse the current networking facilities at DevIS focusing on issues such as: 1) the structure of networking, 2) the problems with networking, and 3) the critical success factors in networking.

On the basis of our analysis of seven networks and the insights on networking from the literature the paper argues that networking efforts should be based on the overall business goals of the SPI unit. The knowledge sharing efforts should be supported by a networking strategy, which addresses the issues related to the practice of knowledge sharing. The networking strategy should be communicated to all practitioners through different levels of the organisation and different types of networking facilities should be developed to support different needs from the SPI unit's different customers. Networking should be an integrated part of practitioners' daily work. The main contribution of the paper to our project was that it helped us understand the current networking practices in our organisation and develop networking facilities to support sharing of personalised knowledge within and outside the SPI unit.

5.6 Summary

Looking at the two purposes of this research (to support the body of knowledge and the software engineering practice at AstraZeneca), we can see that they were both addressed in all of the papers. All papers have contributed to the body of knowledge on SPI and to some extent to the body of knowledge on organisational change and KM (see Figure 4). In parallel, AstraZeneca was already involved in the practice and research that form the basis for each paper, having as its goal to improve the practice of software engineering.

The figure below illustrates the dual nature of the research project and provides an overview of the approaches that were used in producing the five papers presented in this chapter. The research project in general can be approached as a case study based on action research and supported by practice studies and field experiments. The interventions are all based on action research ideas but have been influenced by other approaches as well, as described above and illustrated in figure 10. The SPI practices in this study also followed approaches that can be characterised using the same framework of research approaches.



The SPID Project (Project Level)
Action Research
SPI: Diagnosing and Improving processes
SPI: Analysing and evaluating the SPI project
SPI: Institutionalising the software processes
SPI: Developing KM strategies and mechanisms for managing knowledge
SPI: Analysing and evaluating current networking facilities suggesting networking efforts
Action Research
Academic Paper I
Practice Study
Academic Paper II
Field experiment
Academic Paper III
Practice Study
Academic Paper IV
Practice Study
Academic Paper V

Figure 10. Overview of research approaches

6 DISCUSSION

In this section I discuss the research questions in the light of our findings in the AstraZeneca case and offer some practical advice for how to conduct SPI in practice using insight from knowledge management. I further present areas for further research.

6.1 Making SPI happen in practice

The results of the SPID project have been documented as three industrial reports, three guidelines, 27 templates, better practices, process maps, an Electronic Process Library (EPL) illustrating all processes (presented at AstraZeneca to the steering committee of the SPI project and practitioners, see part III), and five academic papers (presented at different international conferences and journals, see part II) to create new knowledge at AstraZeneca and in academia.

Today (December 2001), the SPID project has been finalised (June 19th of 2001) and the SPI unit has been established (five people) at the company level at AstraZeneca in Mölndal to support practitioners with both customising the new processes and the templates and to work with issues related to quality and compliance in general. The SPI activities are planned and carried out as small improvement cycles in the software organisation led by the SPI unit. The revised versions of the software processes and templates are being used in more than 20 software projects in DevIS and some other IS departments in Mölndal and are appreciated by practitioners working in these software projects. The SPI unit measures and documents the feedback and further improvement suggestions from the practical use of the new processes and templates in these software projects on an ongoing basis. An improvement culture has been cultivated and is maturing in the organisation and we have been encouraged by management to spread the approach of software practice improvement to the whole global organisation.

Our studies illustrate that improving an organisation's software practice is a knowledge management issue. It addresses the current capability of the organisation's software practices, the creation of new processes on the basis of practitioners' ideas, and the modification and transformation of knowledge about the new processes to be implemented and continuously improved in the organisation as a whole. Our experiences suggest that, to better understand SPI and making it happen in practice, we need to approach it as a KM effort and focus more on issues related to improving practices, i.e. Software Practice Improvement. To do this, we need to distinguish between processes and practices and between process knowledge and practical knowledge and study the interaction between these types of knowledge as improvements are made.

Process knowledge is always explicit and can be codified, while practical knowledge is mostly tacit and personalised, although it can also be explicit and codified to some extent. The knowledge creation in SPI occurs when the conversion (see Nonaka and Takeuchi 1995, Arent *et al.* 2001) takes place, from practical knowledge to process knowledge and back again to practical knowledge. Some knowledge will, of course, remain tacit in individual skills and organisational capabilities (see Mathiassen *et al.* 2001). An SPI effort can thus be approached as a KM effort through which both practical knowledge and process knowledge are created, modified, and shared by different actors on different organisational levels, using different approaches (see Arent and Nørbjerg 2000, Arent *et al.* 2001, Mathiassen *et al.* 2001, Pourkomeylian 2001a).

Analysing AstraZeneca's SPI effort from a KM point of view, two related knowledge domains (SPI- and SE-knowledge) are involved during the SPI project's life span. During the initiating phase of the SPI effort, primarily SPI-knowledge was created, modified, and shared on the individual and, sometimes, on the group level. This knowledge was about SPI in general, such as: what SPI is and how it should be organised, planned, and carried out. This knowledge was adapted to the organisation's situation and shared with decision makers and practitioners to create a general understanding about SPI. In the diagnosing phase knowledge about the current maturity level of the software practices was created through an assessment. The results were combined with other findings in earlier quality efforts and completed with suggestions for improvement to be shared with management and other practitioners. In the establishment phase of the SPI effort, mostly SPI-knowledge about resources needed to conduct the SPI activities was created on an individual and sometimes a group level. A shift towards SE-knowledge in the acting phase of our SPI effort took place as new processes were created on the basis of practitioners' ideas and experiences. This knowledge was mainly created on the group level and sometimes on the organisational level through different SPI workshops. Knowledge about the new processes was then transferred to all practitioners on different organisational levels. To facilitate the implementation of the new processes in software projects the processes were modified to fit each software project's specific conditions (size, staffing, task, quality requirements, organisation, and complexity). Finally, in the learning phase, knowledge and experiences from the SPI effort and software processes were documented in order to make this available to all practitioners (see Pourkomeylian 2001a, 2001c).

Practising SPI as a KM effort helped us understand SPI as three parallel activities addressing the creation, modification, and transformation of knowledge in the organisation (see table 6):

1. *Creation*: the objective of the creation activities is to create new or improved SPI- and SE-knowledge. The main actors involved in creating knowledge are: the SPI project manager, management, practitioners, and the SPI expert. The knowledge creation efforts occur mostly on the individual and group levels. Approaches that can be used to facilitate knowledge creation are: assessments, SPI-meetings, SPI-workshops, customisation meetings, and, in some cases, training sessions.
2. *Modification*: the modification activities in SPI are focused on modifying both SPI- and SE-knowledge. The SPI-knowledge should be modified to fit the organisation's conditions (e.g. modify the CMM assessment to fit the organisation's requirements). The SE-knowledge should also be modified to fit the organisation and the individual software projects. The actors involved in the modification efforts are: the SPI project manager, management, the SPI expert, the SPI specialists, and practitioners (project members). The modification efforts occur mostly on the individual and group level (project). SPI-meetings and customisation meetings can be used to structure the modification activities.
3. *Sharing*: the objective of the sharing activities in SPI is to facilitate the communication of both SPI- and SE-knowledge between all organisational levels. The actors involved in this effort are: the SPI project manager, management, practitioners, and SPI specialists. Among approaches that can be used to support knowledge sharing in SPI are: SPI-meetings, group and department meetings, IT-based solutions (to support sharing of codified knowledge), networking facilities (to support sharing of personalised knowledge), customisation meetings, and training sessions.

KM activity	**Objectives**	**Actors**	**Organisational level**	**Approaches**
Creation	Create new or improved Knowledge (SPI- and SE-knowledge)	SPI project manager Management Practitioners	Individual Group (Organisation)	Assessments SPI-meetings SPI-workshops Customisation

		SPI experts		meetings (Training sessions)
Modification	Modify SPI-knowledge to fit the organisation's condition Modify new processes to fit each software project	SPI project manager Management SPI expert SPI specialists Practitioners (project members)	Individual Group (project) (Organisational)	SPI-meetings Customisation meetings
Sharing	Share SPI- and SE-knowledge on different organisational levels	SPI project manager Management Practitioners SPI specialists	Individual Group (project) Organisational	SPI-meetings Group & department meetings IT-based solutions Networking facilities Customisation meetings Training sessions

Table 6. The KM process in SPI

In summary, SPI is about managing software knowledge. To make SPI happen in practice we need to focus not only on processes, but equally important on issues related to improving practices. To do this, we need to distinguish between processes and practices and between process knowledge and practical knowledge and study the interaction between these types of knowledge as expressed in the creation, modification, and sharing of knowledge across individual, project, and organisational levels.

6.2 Knowledge Management challenges in SPI

Facilitating the interaction efforts and making it happen is crucial for succeeding in creating knowledge in SPI (Arent and Nørbjerg 2000, Pourkomeylian 2001a). The quality of the knowledge created is dependent on the quality of practitioners' knowledge about the software practices in the organisation and the success of the conversion from practical knowledge to process knowledge (Nonaka and Takeuchi 1995, Arent and Nørbjerg 2000, Arent *et al.* 2001, Mathiassen *et al.* 2001). It is important that people with relevant competence and experience participate in SPI-workshops. Organising, planning, and gathering people for SPI-workshops is a time-consuming process. It is difficult and to a certain extent impossible to capture practitioners' knowledge about practices within a focused area (e.g. software change control), agree upon a common understanding about how to do change control, and externalise such insights in the form of explicit knowledge (process knowledge). Another challenge appears through implementation efforts, where the explicitly documented knowledge is adapted and used by practitioners in practice. There is a gap between what is described in a guideline and implementing the actions in practice through modification and adaptation of knowledge. The modification effort is iterative, leading to new knowledge to be used in the specific projects (see Arent and Nørbjerg 2000, Pourkomeylian 2001b, Arent *et al.* 2001, Pourkomeylian 2001c).

In our experience one key factor and a great challenge in SPI is knowledge sharing. Different types of knowledge should be shared between actors on different organisational levels (Kautz and Nielsen

2001, Pourkomeylian *et al.* 2001, Mathiassen and Pourkomeylian 2001). If the knowledge sharing activities are not addressed in an SPI project as planned activities there is a risk that the knowledge sharing issues will not be addressed at all. This might cause problems in gaining management and practitioner commitment and acceptance, for the results of the project and for the SPI effort as a whole.

To improve practice, software organisations must learn about SPI (Kautz and Nielsen 2001). Several studies argue that routines and procedures are learnt and developed in practice, and not solely from formal (explicit) procedures (Brown and Duguid 1991, Mathiassen *et al.* 1997). In our experience learning SPI is about creating, adapting, and transferring software knowledge to different organisational levels. From this point of view, SPI initiatives should emerge through personal growth, through knowledge creation, through knowledge adaptation, and through knowledge transformation to first reach internalisation in practice (see Pourkomeylian 2001c).

To make it happen one strategy can be to support learning by sharing knowledge and learning through reflection-in-action (see Schön 1987). This is what Arent *et al.* (2001) call the exploration strategy, which is focused on changing practice by creating personalised knowledge. This strategy should be initiated at the project level, in which individuals learn new ways of working through observation or imitation. It requires some mechanisms to collect and share experiences, to identify better practices, and subsequently to transfer this knowledge across the organisation. This strategy results in quick but still uncoordinated improvements at the project level and provides weak support for organisational learning (Arent *et al.* 2001). In AstraZeneca's case we combined process knowledge with knowledge about the specific conditions in each software project to create new practices in software projects (through customisation meetings). This was experienced as difficult and time-consuming, especially when a project team goes through a customisation process for the first time. The technical complexity of projects in combination with the complexity of quality rules often forced us to hold several meetings to come to a joint decision. This process became easier when people had experience from earlier customisation meetings. In these meetings we encouraged practitioners to create variations on the basis of the general software processes and reflect-in-their-actions and adapt the customised processes during the project's life span (Pourkomeylian 2001c).

The other strategy is based on learning by reflection and integration through documenting practice. This is creation of explicit knowledge across the organisation in the form of new procedures or guidelines. This is what Arent *et al.* (2001) call the exploitation strategy. There are some risks involved with this strategy, e.g. it delays actual changes in software projects. Further, there is no guarantee that the new processes will actually be useful when finally implemented. One way to avoid these risks is to release unfinished processes as early as possible and allow for fast publication and refinement through use. This means that the SPI unit must be prepared to improve and refine the new processes based on experiences from practice. The SPI unit must also set up feedback channels and plan for rewriting and republishing new guidelines for the processes. This is a very time–demanding process (see Pourkomeylian 2001c). The strength of this strategy is its ability to produce explicit knowledge in the form of procedures and guidelines across the organisation by reflection on and integration of existing practices. IT-based solutions (in our case the EPL) can be used to facilitate sharing of this knowledge between different organisational levels (see Mathiassen and Pourkomeylian 2001). Our SPI effort included both exploration and exploitation strategies to facilitate the knowledge sharing efforts in the organisation (see Pourkomeylian 2001c, Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001).

In summary, one key factor and a great challenge in SPI is knowledge sharing. SPI initiatives should emerge through personal growth, through knowledge creation, through knowledge adaptation, and through knowledge transformation to first reach internalisation in practice. It is difficult and to a certain extent impossible to capture practitioners' knowledge about practice. The focus should rather be on capturing a common agreement on the "what" issues (e.g. what do we mean by project planning?) and a general framework for the "how" issues (e.g. how should we do project planning?). Modification of processes in each project is a time-consuming process but a very effective effort for implementing new processes in practice. Developing different knowledge sharing strategies and mechanisms is a time-demanding task and should be planned early in the SPI project.

6.3 Using insights from Knowledge Management in SPI

We used the concept of Nonaka and Takeuchi (1995) to understand and address the knowledge creation and modification issues in SPI. To support knowledge sharing between different organisational levels we used Hanssen *et al.'s* (1999) concept of codified and personalised knowledge. The knowledge sharing efforts in our project were guided by different strategies and mechanisms depending on the different types of SPI services to support both codified and personalised knowledge (see Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001). Because knowledge is basically hermeneutic and only partially supported by positivistic elements, a KM strategy for SPI should be expected to focus on the personalised approach. This was not entirely the case at AstraZeneca, however. We considered differences between the services provided by our SPI unit and adopted specialised strategies based on the characteristic features of our services. We believe that each SPI unit has to find its own balance between the personalised and codified approaches. This balance needs to be dynamically adjusted as the SPI unit matures. Considering the different nature of the IDEAL model and the CMM we suggested that improvement and innovation oriented services should be based mainly on personalised approaches whereas maintenance and diffusion of processes, procedures, and templates are more open to approaches that are essentially codified. This can be explained as follows.

The CMM is in itself codified knowledge of software practices presented and described on different levels of detail, with goals and key practices for each key process area. The very idea is to reuse this knowledge across software organisations to assess capabilities and to guide action. To reach level 2 (repeatable), organisations are advised to implement specific key process areas. The new processes should, however, not be implemented in a standardised fashion. Rather, it is recommended that each software project finds its own way to reach the goals expressed in CMM. The strategy is in this phase therefore mainly personalised and supplemented with some codified knowledge from CMM and state-of-the-art SE-knowledge. To continue further towards level 3 (defined), the focus changes to a standardisation of software processes across the software organisation. Processes have to be explicitly defined and maintained and new meta-processes should be created to support their adaptation to specific projects. As the organisation approaches level 3 the main KM strategy is codified and even the customisation of this knowledge to specific software projects is itself codified (see Mathiassen and Pourkomeylian 2001).

In the case of AstraZeneca, our goal was not to reach any particular level in the CMM. We focused on learning SPI, on improving (creating) and implementing a few software processes in the organisation, and on increasing practitioners' understanding about company quality rules and requirements. As a pharmaceutical company we must have SOPs (written documents) in place that address quality regulations in software projects. In this sense, most of the knowledge created in our project had the character of being codified (written documents). In addition, we aimed to develop and implement process knowledge across the organisation. To do this we adopted a KM strategy based on a primary, codified approach and a supplementary, personalised approach to support sharing of codified knowledge at DevIS.

We codified knowledge that should and could be codified (such as guidelines, procedures or quality rules). IT-based solutions were used to support fast accessibility and availability of the codified knowledge on different organisational levels to all practitioners. The codified knowledge was structured and presented in a way that, in combination with IT, supported learning (see Pourkomeylian 2001c). To support the sharing of knowledge that cannot or should not be codified (e.g. how to customise the newly created processes to fit to software projects) we established networking facilities on the basis of a personalised strategy. Hence, we created a framework for software processes using a codified strategy and focused on and supported software practices using a personalised strategy.

Codifying, structuring, and presenting knowledge using IT-based solutions demands a great deal of time and resources. The IT-based solution is a software application that should be developed and maintained. If these activities are not planned in the SPI project from the beginning, raising extra resources in the middle of the project may be a tough task. Establishing networking facilities to support sharing of knowledge about practices can also be problematic if the activity is not based on an organised and planned networking strategy. The networking efforts should be based on the organisation's business goals and supported by management (see Mathiassen and Pourkomeylian 2001, Pourkomeylian *et al.* 2001).

In summary, we have found it useful to let SPI projects focus on the SPI and KM concepts rather than on model-based recommendations like those of the CMM. Further, the projects should aim to understand the most characteristic features of the organisation in question and improve a few processes in a stepwise manner. The SPI project should have a project specification including plans, schedules, deliverables, resources, roles and responsibilities and an overall KM strategy. A MAP analysis can be used to facilitate understanding the most characteristic features of the organisation's SPI effort (see Aaen *et al.* 2001, Mathiassen *et al.* 2001, Pourkomeylian 2001b). To facilitate the creation, modification, and sharing of knowledge in the SPI project, a KM strategy should be developed early in the project. It should be based on the organisation's business goals and address issues related to both codified and personalised knowledge.

6.4 Further Research

Other studies have argued for and illustrated the usefulness of applying KM to SPI (see Baskerville and Pries-Heje 1999, Arent and Nørbjerg 2000, Kautz and Nielsen 2001, Kautz and Thaysen 2001, Mathiassen *et al.* 2001, Arent *et al.* 2001). What we did in addition to these studies was to systematically explore on a practical level how an SPI initiative could be informed by applying KM insights as an integral part of the effort. Our studies have, as described above confirmed that for making SPI happen in practice it is useful to apply KM insights in SPI. Doing so provides a deeper understanding of some of the key challenges involved and it guides actions in a number of useful ways. The key contribution of our studies is that it provides additional insights of how this can be achieved on a practical level and that it points in direction of further research within this area.

Capturing practitioners' practical knowledge about practice is not an easy task. To address this issue we used different approaches like SPI-meetings, SPI-workshops, customisation meetings, and in some cases training sessions. This was a difficult and rather time-demanding effort (see Pourkomeylian 2001a, 2001b, 2001c). This gives rise to further questions: How can we make this process more efficient? How do we know that we have captured enough and relevant knowledge?

Once new knowledge is created, one great challenge is transferring knowledge to different organisational levels. Codification has been suggested as a strategy to facilitate this effort. Interesting issues for further studies are: Which knowledge should be codified? What techniques should be used? How to illustrate codified knowledge?

Not surprisingly, codification is not the answer to all knowledge sharing problems. A personalised strategy can be used to address issues related to transformation of personalised knowledge. But, which knowledge is personalised knowledge? Where is the borderline between codified and personalised knowledge in SPI and how to recognise this line?

In this study we have used the IDEAL model to structure our SPI effort, the CMM model to focus on specific areas, and the MAP framework to understand the nature of SPI. These models do not, however, explicitly address the practical issues of managing knowledge in SPI. The following question can be the point of departure for further research:

- How to address KM issues in CMM, IDEAL and the MAP framework to support practical issues of creation, modification, and knowledge sharing in SPI?

My ambition in this thesis has not been to provide complete coverage to make SPI more effective by implementing KM insights, but rather to understand the complexity of SPI by approaching it from a KM perspective in practice. I hope to have opened a practical and theoretical discussion to stimulate further investigations of KM as a facilitator for improving SPI practice in AstraZeneca and in other software organisations.

REFERENCES

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement. *Scandinavian Journal of Information Systems,* Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 123-146.

Ami (1992). *Ami, application of metrics in industry, A Qualitative Approach to Software Management.* CSSE South Bank University, London, pp. 1-35.

Andersson, I. and Nilsson, K. (2001). Diagnosing Diffusion Practices within a Software Organisation. *IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Arent, J. and Iversen, J. (1996). *Development of a Method for Maturity Assessments in Software Organizations based on the Capability Maturity Model.* Aalborg University: Department of Computer Science.

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science.*

Arent, J., Pedersen, M. H. and Nørbjerg, J. (2001). Strategies for Organizational Learning in SPI. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice,* Addison-Wesley.

Augier, M. and Vendelo, M. T. (1999). Networks, Cognition and Management of Tacit Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 252-261.

Bach, J. (1994). The Immaturity of the CMM. *American Programmer*, Vol. 7, No. 9, pp. 13-18.

Baskerville, R. and Pries-Heje, J. (1999). Managing Knowledge Capability and Maturity. In: Larsen, T., J., Levine, L., and DeGross, J., I. (Eds.): *Information Systems: Current Issues and Future Change.* Norwell, MA: IFIP/Kluwer Academic Publisher.

Baskerville, R. and Wood-Harper, A. T. (1996). A Critical Perspective on Action Research as a Method for Information Systems Research. *Journal of information technology*, Vol. 11.

Brown, J. S. and Duguid, P. (1991). Organisational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning and Innovation. *Organization Science*, Vol. 2, No. 1, pp. 40-57.

Cook, S. D. N. and Brown, J. S. (1999). Bridging Epistemologies: The Generative Dance between Organisational Knowledge and Organisational Knowing. *Organisation Science*, Vol. 10, No. 4, pp. 381-400.

Dahlbom, B. and Mathiassen, L. (1993). *Computer in Context, The Philosophy and Practice of System Design*. Blackwell Publishers.

Dick, B. (1997a). Action learning and action research. [On line] available at: http://www.scu.edu.au/schools/sawd/arr/actlearn.html

Dick, B. (1997b). Action research FAQ: “frequently asked questions” file. [On line] available at: http://www.scu.edu.au/schools/sawd/arr/arfaq

Dick, B. (1997c). A beginners guide to action research. [On line] available at: http://www.scu.edu.au/schools/sawd/arr/guide.html

Debou, C. (1997). SPI Success Factors: Toward More Business Orientation. *Software Process Newsletter, IEEE Computer Society,* No. 10, pp. 15-18.

Galliers, R. D. (1992). Choosing an Information Systems Research Approach. *Information Systems Research: Issues, Methods, and Practical Guidelines,* R. D. Galliers, ed., Blackwell Scientific Publications, Oxford, pp. 144-162.

Galliers, R. D. and Land, F. L. (1987). Choosing Appropriate Information Systems Research Methodologies. *Communications of the ACM,* Vol. 30, No. 11, pp. 900-902.

Goldenson, D. R. and Herbsleb, J. D. (1995). *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-009.

Halloran, P. (1999). Organizational Learning from the Perspective of a Software Process Assessment and Improvement Program. *Proceedings of Hawaii International Conference on Systems Science*, Los Alamitos, CA: IEEE Computer Society Press.

Hansen, T., Morten, N. and Tierney, T. (1999). What’s Your Strategy for Managing Knowledge? *Harvard Business Review*, pp. 106-116.

Hayes, W. and Zubrow, D. (1995). *Moving On Up: Data and Experience Doing CMM-Based Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-008.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. C. (1997). Software Quality and the Capability Maturity Model. *Communications of the ACM*, Vol. 40, No. 6, pp. 30-40.

Humphrey, W. (1989). *Managing the Software Processes*. Addison-Wesley.

Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991). Software Process Improvement at Hughes Aircraft. *IEE Software*, Vol. 8, No. 4, pp. 11-23.

Johansen, J. and Mathiassen, L. (1998). Lessons learned in a National SPI Effort. EuroSPI 98 Gothenburg, Sweden, November 16-18, pp. 5.1-17.

Kautz, K. and Nielsen, P. A. (2001). Knowing and Implementing SPI. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice*, Addison-Wesley.

Kautz, K. and Thaysen, K. (2001). Knowledge, Learning and IT Support in a Small Software Company. *Proceedings of the European Conference on Information Systems*, Bled, Slovenia.

Kuvaja, P. and Bicego, A. (1994). BOOTSTRAP - a European assessment methodology. *Software Quality Journal*, Vol. 3, pp. 117-127.

Land, F. F. (1986). *Social Aspects of Information Systems. In Management Information Systems: The Technology Challenge*. N. Piercy, Ed. Croom Helm.

Larsen, E. Å. and Kautz, K. (1997). Quality Assurance and Software Process Improvement in Norway. *Software Process - Improvement and Practice*, Vol. 3, No. 2, pp. 71-86.

Mashiko, Y. and Basiili, V. R. (1997). Using the GQM Paradigm to Investigate Influential Factors for Software Process Improvement. *Journal of Systems and Software*, Vol. 36, pp. 17-32.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

Mathiassen, L., Munk-Madsen, A., Nielsen, P. A. and Stage, J. (1997). *"Methods" in Reflective System Development.* Vol. 2. Aalborg University: Department of Computer Science, pp. 349-365.

Mathiassen, L. and Pourkomeylian, P. (2001). Knowledge Management in a Software Process Improvement Unit. *International Conference on Managing Knowledge: Conversations and Critiques, University of Leicester, England.*

Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.) (2001). *Improving Software Organisations - From Principles to Practice.* Addison-Wesley.

McFarlan, F. W. (1984). *The Information Systems Research Challenges.* Harvard Business School Press.

McFeeley, B. (1996). *IDEAL. A User's Guide for Software Process Improvement*. The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Handbook CMU/SEI-96-HB-001.

Mingers, J. and Stowell, F. (1997). *Information Systems, An Emerging Discipline?* McGraw-Hill.

Mingers, J. (2001). Combining IS Research Methods: Towards a Pluralist Methodology. *Information Systems Research*, Vol. 12, No. 3, September 2001, pp. 240-259.

Mumford, E., Hirschheim, R. A., Fitzgerald, G. and Wood-Harper A. T. (Eds.) (1986). *Research Methods in Information Systems.* Amsterdam, North Holland.

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

Olson, T., Humphrey, W. and Kitson, D. (1989). *Conducting SEI-assisted Software Process Assessments.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-90-TR-7.

Patton, M. Q. (1990). *Qualitative Evaluation and Research Methods*. Sage Publications.

Paulk, M. (1996). Effective CMM-Based Process Improvement. *6th International Conference on Software Quality*, Ottawa, Canada.

Paulk, M., Curtis, B., Chrissis, M. and Weber, C. (1993). *Capability Maturity Model for Software Version 1.1.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-96-TR-24.

Pressman, R. S. (1997). *Software Engineering, a practitioner's approach*. Fourth edition, McGraw-Hill, International Editions, Software Engineering Series.

Pourkomeylian, P. (2001a). Knowledge Creation in Improving a Software Organization. *Proceedings from IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Pourkomeylian, P. (2001b). Analysing an SPI Project with the MAP Framework. *Scandinavian Journal of Information Systems*, Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 147-165.

Pourkomeylian, P. (2001c). An Approach to Institutionalisation of Software Processes. *Proceedings from Tenth International Conference on Information Systems Development*, London, England.

Pourkomeylian, P., Hörnell, J. and Söderberg, S. (2001). Knowledge Sharing in a Software Process Improvement Unit. *Proceedings from The Second European Conference on Knowledge Management*, Bled, Slovenian.

Scarbrough, H. and Swan, J. (1999). Knowledge Management and the Management Fashion Perspective. *Proceedings from, The British Academy of Management Conference on Managing Diversity*, Manchester, England.

Scarbrough, H. and Swan, J., ed. (1999). *Case Studies in Knowledge Management*. London: Institute of Personnel and Development.

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

Schön, D. A. (1987). *Educating the Reflective Practitioner*. Jossey-Bass Publishers.

Seufert, A., Bach, A. and Von Krogh, G. (1999). Towards Knowledge Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 183-185.

Shariq, S. Z. (1999). How does Knowledge Transform as it is Transferred? Speculations on the Possibility of a Cognitive Theory of Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 243-251.

Stelzer, D., Mellis, W. and Herzwurm, G. (1998). Technology Diffusion in Software Development Processes: The contribution of Organizational Learning to Software Process Improvement. In Larsen, T. and McGuire, E. (Eds.): *Information Systems Innovation and Diffusion: Issues and Directions*, Idea Group Publisher.

Swan, J., Newell, S. and Robertson, M. (2000). Limits of IT-Driven Knowledge Management Initiatives for Interactive Innovation Processes: Towards a Community-Based Approach. *Proceedings of the 33rd Hawaii International Conference on System Sciences*.

Swan, J., Newell, S., Scarbrough, H. and Hislop, D. (1999). Knowledge Management and Innovation: Networks and Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 262-275.

TickIT (1995). *TickIT, Guide to Software Quality Management System Construction and Certification using ISO 9001:1994.* Issue 1.0, DISC TickIT Office.

Thomson, H. E. and Mayhew, P. (1997). Approaches to Software Process Improvement. *Software Process-Improvement and Practice*, Vol. 3, No. 1, pp. 3-17.

Tsoukas, H. (1996). The Firm as Distributed Knowledge System: A Constructionist Approach. *Strategic Management Journal*, Winter Special Issue, Vol. 17, pp. 11-25.

Vogel, D. R. and Wetherbe, J. C. (1984). MIS research A profile of leading journals and university. *Data Base* 16, 3.

Wohlwend, H. and Rosenbaum, S. (1994). Schlumberger's Software Improvement Program. *IEEE Transactions on software engineering,* Vol. 20, No. 11, pp. 833-839.

Zack, M. H. (1999). Developing a Knowledge Strategy. *California Management Review,* Vol. 41, No 3, pp. 125-145.

Zahran, S. (1998). *Software Process Improvement: Practical Guidelines for Business Success.* Addison-Wesley.

II RESEARCH CONTRIBUTIONS

The versions of the papers included in this thesis are, except for minor corrections identical to the published papers.

KNOWLEDGE CREATION IN IMPROVING A SOFTWARE ORGANISATION

Pouya Pourkomeylian

Abstract: Software Process Improvement (SPI) is a systematic approach for improving the capabilities of a software organisation. This study shows the results of a collaborative research initiative in which an SPI project was conducted and analysed as organisational knowledge creation. The study explains how knowledge is created through transformation between tacit and explicit knowledge and through interaction between different organisational levels of actors. On the basis of our findings it is suggested that two types of knowledge are created in an SPI project based on completely different knowledge creation behaviour.

Keywords: Software Process Improvement (SPI), Organisational Knowledge Creation (OKC).

1 INTRODUCTION

Software development has existed as a discipline for more than forty years but has not yet become a disciplined process. Software projects are almost always later than expected, the costs of developing software are higher than planned, and the functionality and the quality of the final products (software and documentation) are lower than expected. Software organisations have used different methods for improving software processes. The most recent approach for improving software processes is Software Process Improvement (SPI), which is a systematic approach toward changing software development practice.

The first step in improving software processes is to understand the current status of the software development process (Humphrey 1989). One way of doing this is to make an assessment based on a model as a road map. During the past years software organisations have used different appraisal approaches to identify what should be improved in their software processes. The most popular assessment model is the Capability Maturity Model (CMM), which is a normative approach to software process improvement developed by the Software Engineering Institute (SEI) (Paulk 1993). Other approaches include BOOTSTRAP (Kuvaja 1994), SPICE (Thomson and Mayhew 1997), ami (ami 1992), TickIT (TickIT 1995), and TRILLIUM (Thomson and Mayhew 1997). Common for all these approaches is that they apply Total Quality Management (TQM) principles to SPI. After making an assessment further improvement activities should be planned and performed to create new or modified software processes.

Different reports have pointed out difficulties in performing SPI projects in practice (Goldenson and Herbsleb 1995, Debou 1997). An SPI effort is successful when the new or modified software processes are created and used in the organisation's daily practice and have been proven to function in achieving their goals. Success with SPI seems to depend on a complex mix of highly interrelated factors acting in different phases in an SPI project. Different factors such as scaling the SPI initiative, setting realistic goals, the complexity of organisational changes, and the organisational culture have made it difficult to achieve success in SPI initiatives (Goldenson and Herbsleb 1995, Herbsleb *et al*. 1997, Mashiko and Basili 1997, Johansen and Mathiassen 1998). But SPI has also been shown to be able to help organisations gain organisational benefits (Wohlwend and Rosenbaum 1994, Hayes and Zubrow 1995, Larsen and Kautz 1997).

An organisation's software development practices are based on the existing knowledge of practitioners and mana gers about the software development practice (Arent and 2000). To change software development practices the organisation should improve the practitioners' existing knowledge (both tacit and explicit) of the software practices. The created new or modified knowledge should then be transferred to all organisational levels to become part of the practitioners' daily work. Creating new or modified software processes in this way is a knowledge Nørbjerg creation process in which different actors at different organisational levels are involved in creating different types of knowledge.

Some recent reports have reflected on the importance of creating and managing knowledge and learning issues for SPI initiatives. Arent and Nørbjerg (2000) analysed how organisational knowledge creation process and learning can support SPI initiatives. Stelzer *et al.* 1998 studied how principles and technologies from organisational learning can apply to SPI initiatives and become enablers of SPI success. Halloran (1999) investigated the relationship between an SPI approach and organisational learning. These studies indicate that the concept of knowledge creation and learning can support SPI initiatives. We believe that the concept of organisational knowledge creation has much to offer the SPI community, especially in the following three questions: 1) What types of knowledge are created as the result of performing an SPI project? 2) Which actors are involved in the knowledge creation process? 3) How do they interact to create knowledge?

As a framework to support the analysis of the SPI project this study has chosen Nonaka and Takeuchi's theory (see Nonaka and Takeuchi 1995) because their theory explicitly deals with the fundamental process of knowledge creation and supports an understanding of the interaction between individuals, groups, and organisations in the knowledge creation process. This approach has demonstrated its usefulness in relation to SPI in a study by Arent and Nørbjerg (Arent and Nørbjerg 2000) analysing the learning process in SPI.

1.1 The Research Approach

Using a collaborative practice research approach (see Mathiassen 2000) this study has combined *action research* in combination with *field experiment*, and *practice study* aiming to change practice. Improving practice is the distinguishing feature of collaborative practice research and action research in general (Baskerville *et al.* 1996, Mathiassen 2000). The action research approach is chosen for this study because of its strong support in: 1) integrating research and practice, 2) involving practitioners in the problem being studied, 3) giving the possibility of introducing change at the same time the research is going on.

In this study an SPI project was done during the period of April 1999 to June 2000. Several practitioners (software engineers) were involved during both the evaluation of software projects and the improvement of software processes. The SPI-group including: software engineers, assisting consultants, and the author (leading the SPI project) working with improving software processes became a forum for evaluating the Software Engineering (SE) and the SPI practices, for creating and experimenting with new or modified software processes, and for learning about SPI in practice. Action research in this study was set up to improve three main software processes using the IDEAL model (see section 2.1). Field experiments were set up as controlled research efforts in which the created software processes were tested in one selected software pilot project to show the effects of the created processes. Focused practice studies were initiated to learn about the current maturity level of the software organisation.

One focused SPI practice was to make a CMM assessment to establish the current maturity of software processes. To collect data about the current capability of software processes at the software organisation a modified CMM assessment based on a method called Questionnaire Based Assessment (QBA) (see Arent and Iversen 1996) was made for three different software development projects chosen from two different software development groups. The following Key Practice Areas (KPAs) were included in the assessment to identify software process problems: 1) Software Project Planning, 2) Requirements Management, 3) Software Project Tracing and Oversight, 4) Software Quality Assurance, 5) Software Configuration Management.
Project managers and developers of three selected software development projects answered the CMM-based questionnaire. The collected data were statistically analysed and proposals were developed for improving software development projects on the basis of the results of analysed qualitative data collected from the assessment, the software process improvement literature, and other quality improvement findings from earlier quality activities in the software organisation. The SPI-group met at least eight times throughout the 14-month period for planning and organising SPI initiatives and discussing difficulties and problems. The new and modified software processes will be implemented in the whole organisation starting in August 2000. The results and lessons learned during the improvement phase have been documented. These lessons have been both interpretative, i.e. helped us to understand the practice, and normative, i.e. helped us to design new or modified software processes and improve the practice. Knowledge gained and experience from doing this research in practice have created new research activities for further studies. The author actively participated in the practical work with SPI in the organisation, such as conducting the CMM-based assessment, running and participating in workshops and seminars, performing interviews and analysing results.

The section below discusses the software process improvement and organisational knowledge creation concepts and presents the framework for the analysis. Section 3 presents the case. Section 4 presents a map of the knowledge creation process in the SPI project and discusses the findings according to the three questions mentioned above, and section 5 concludes the paper by presenting the lessons learned and pointing out areas for further research.

2 BACKGROUND

Software Process Improvement (SPI) was originally developed at the Software Engineering Institute (SEI) at the Carnegie Mellon University and was based on ideas presented by Humphrey (see Humphrey 1989). According to Aaen *et al*. (2001) SPI is based on a number of ideas that offer answers to specific concerns. SPI has three fundamental concerns: the *management* of SPI activities, the *approach* taken to guide the SPI initiatives, and the *perspective* used to focus attention on the SPI goal(s).

The management of SPI initiatives is based on three ideas: 1) the SPI activities are *organised* in a dynamic fashion, 2) all improvement efforts are carefully *planned* and 3) *feedback* on effects on software engineering practices is ensured. The approach to SPI initiatives is guided by three additional ideas: 1) SPI is *evolutionary* in nature, 2) SPI is based on idealised, *normative* models of software engineering and 3) SPI is based on a careful creation and development of *commitments* between the involved actors. Finally the perspective forward the SPI goal is dominated by three ideas: 1) SPI is focused on *software processes*, 2) the practitioners' *competencies* are seen as the key resource and 3) SPI aims to change the *context* of the software operation to create sustainable support for the actors involved. The basic idea in SPI is to focus on software processes as social institutions with a complex interplay of people, methods, tools and products (Aaen *et al*. 2001). SPI is focused on improving software processes based on practitioners' ideas and experiences. This involves capturing practitioners' tacit knowledge (know-how) and transferring it to explicit knowledge, which should then be combined with the organisation's other explicit knowledge prepared for use in practice by all practitioners in different organisational levels.

2.1 The IDEAL Model

A popular model in the field of SPI that is suitable for assistance in managing SPI initiatives for implementing organisational changes is the IDEAL model (see McFeeley 1996). As shown in figure 1 the IDEAL model considers five phases (Initiating, Diagnosing, Establishing, Acting and Learning) of a software process improvement initiative, which provide a continuous loop through the steps necessary for software process improvement (McFeeley 1996).

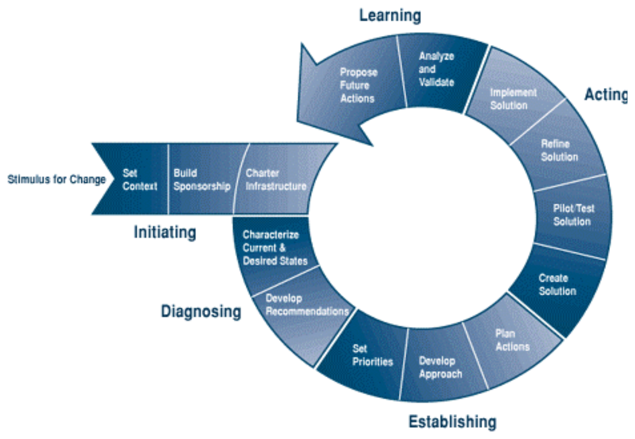


Figure 1. The IDEAL Model (McFeeley 1996)

Once the first cycle of SPI has been completed there will be a need to regularly repeat the entire process. However, the ultimate goal in organisations should be to succeed in achieving process implementation in the whole organisation. This should lead to the creation of a process culture in which process discipline prevails. Our intention in using the IDEAL model was to establish successful improvement activities and infrastructures for SPI initiatives within the software organisation. This study includes the Initiating, Diagnosing, Establishing, and some parts of the Acting and Learning phases.

2.2 Organisational Knowledge Creation

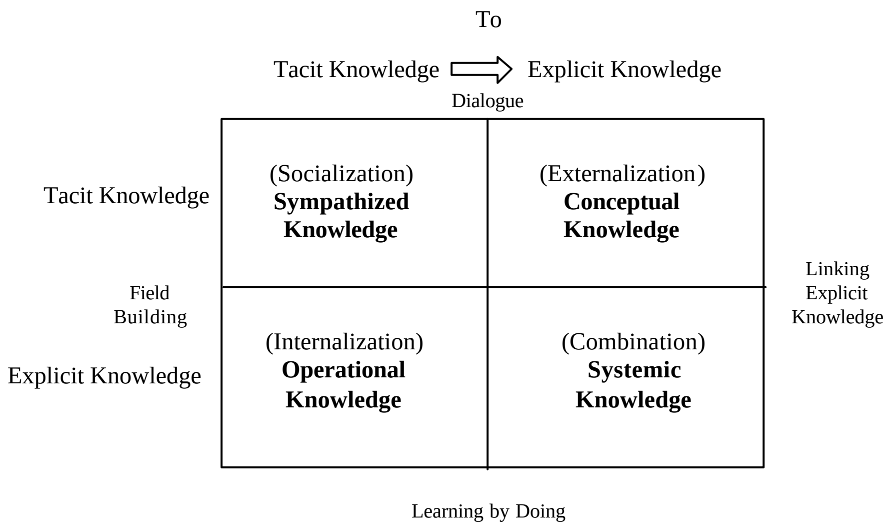
Nonaka and Takeuchi (1995) use two dimensions of knowledge creation to explain the process of organisational knowledge creation: 1) *the ontological* and 2) *the epistemological*.
The *ontological* dimension focuses on individual knowledge creation. The organisation supports creative individuals or provides a context for them in which to create knowledge. Organisational knowledge creation is understood as a process that "organisationally" amplifies the knowledge created by individuals and crystallises it as a part of the knowledge network of the organisation. This process takes place within an expanding "community of interaction", which crosses intra- and inter-organisational levels and boundaries.

For the *epistemological* dimension Nonaka and Takeuchi (1995) draw on Michael Polanyi's (1966) distinction between explicit knowledge and tacit knowledge. Explicit knowledge refers to knowledge that is transmittable in formal, systematic languages. It can be articulated in formal languages including grammatical statements, mathematical expressions, specifications, manuals and so forth. It can be transmitted across individuals formally and easily. Tacit knowledge is personal, context-specific, and therefore difficult to formalise and communicate. It is personal knowledge embedded in individual experience and involves intangible factors such as personal belief, perspective, and the value system. Tacit knowledge is difficult to communicate and share in the organisation and must thus be converted into words or numbers that anyone can understand. Nonaka and Konno (1998) describe two dimensions to tacit knowledge. The first dimension is the *technical* dimension, which encompasses the kind of informal personal skills or crafts often referred to as "know-how". The second dimension is the *cognitive* dimension, which consists of beliefs, values, ideals and mental models that are deeply ingrained and which we often take for granted. They argue further that this cognitive dimension of tacit knowledge shapes the way we perceive the world. This kind of knowledge could also be defined as procedural knowledge used in problem solving and decision making (Nonaka and Takeuchi 1995, Firebaugh 1989). According to Nonaka and Takeuchi (1995) organisational knowledge is created during the time the "*conversion*" takes place, i.e. from tacit to explicit and back again into tacit. The interaction between these two forms of knowledge is the key dynamic of knowledge creation in the organisation.

Knowledge Conversion

Knowledge conversion is a "social" process between individuals and is not confined to one individual. Assuming that knowledge is created through the interaction between tacit and explicit knowledge four different modes of knowledge conversion are possible (Figure 2). The content of the knowledge created by each mode of knowledge conversion is naturally difficult, which creates different contents of knowledge (Nonaka and Takeuchi 1995):

1. From tacit knowledge to tacit knowledge (socialisation that creates sympathised knowledge). The socialisation mode usually starts with building a "field" of interaction. This field facilitates the sharing of members' experiences and mental models. Socialisation involves the sharing of tacit knowledge between individuals.
2. From tacit knowledge to explicit knowledge (externalisation that creates conceptual knowledge). The externalisation mode is triggered by meaningful "dialogue or collective reflection," in which using appropriate metaphor or analogy helps team members to articulate hidden tacit knowledge that is otherwise hard to communicate. Externalisation requires the expression of tacit knowledge and its translation into comprehensible forms that can be understood by others.
3. From explicit knowledge to explicit knowledge (combination that creates systematic knowledge). The combination mode is triggered by "networking" newly created knowledge and existing knowledge from other groups in the organisation, thereby crystallising them into a new product or service.
4. From explicit knowledge to tacit knowledge (internalisation that creates operational knowledge). "Learning by doing" triggers internalisation. The internalisation of newly created knowledge is the conversion of explicit knowledge into the organisation's tacit knowledge. In practice, internalisation relies on two dimensions. First, explicit knowledge must be embodied in action and practice. Second, there is a process of embodying the explicit knowledge by using simulations or experiments to trigger learning by doing processes.



To
Tacit Knowledge
Explicit Knowledge
Dialogue
Tacit Knowledge
(Socialization) Sympathized Knowledge
(Externalization) Conceptual Knowledge
Field Building
Linking Explicit Knowledge
Explicit Knowledge
(Internalization) Operational Knowledge
(Combination) Systemic Knowledge
Learning by Doing

Figure 2. Contents of knowledge created via the four modes (Nonaka and Takeuchi 1995)

This study is focused on the *process* issues of knowledge creation in SPI, the actors involved, and the conversion of new or modified knowledge between different organisational levels. We interpret changes in practitioners' understanding of the software processes, and changes in practice as indicators of new *tacit knowledge,* and new guidelines, policies, and manuals as new *explicit knowledge*.

3 THE CASE

This study was performed at AstraZeneca one of the world's leading pharmaceutical companies. AstraZeneca is a research-driven organisation with a range of products designed to fight disease in different areas of medical need. The company was formed in April 1999 by the merger of Astra AB and Zeneca Group PLC. AstraZeneca has a strong research base and powerful product portfolio, designed in seven areas of true medical need – cancer, cardiovascular, central nervous system, gastrointestinal, infection, pain control and, and respiratory. AstraZeneca is globally number three (1999) in ethical pharmaceuticals and has more than 50,000 employees worldwide. It has research and development (R&D) centers in Sweden, UK and the USA and R&D headquarters in Södertalje, Sweden. The company has some 10,000 R&D personnel and a US $2 billion R&D investment in 1999, extensive global sales and marketing network, employing over 25,000 people, and 12,000 people employed in production in 20 countries.

3.1 The Software Organisation

AstraZeneca has four departments that supply global IT services to the whole company: one in the UK, one in the USA, one in Sweden and one to provide IT support for research and development for the whole organisation. In addition to this, there are five global supplier managers who have the responsibility of controlling the needs of IT services in the business functions in the company. Furthermore, there is a company staff with central IT departments for solving problems related to technology adoptions, infrastructure, security, integration, and strategies. Beyond all this, there are IT functions that support the local marketing company in the respective countries. There are in total 2,500 persons working with IT-related questions in AstraZeneca.

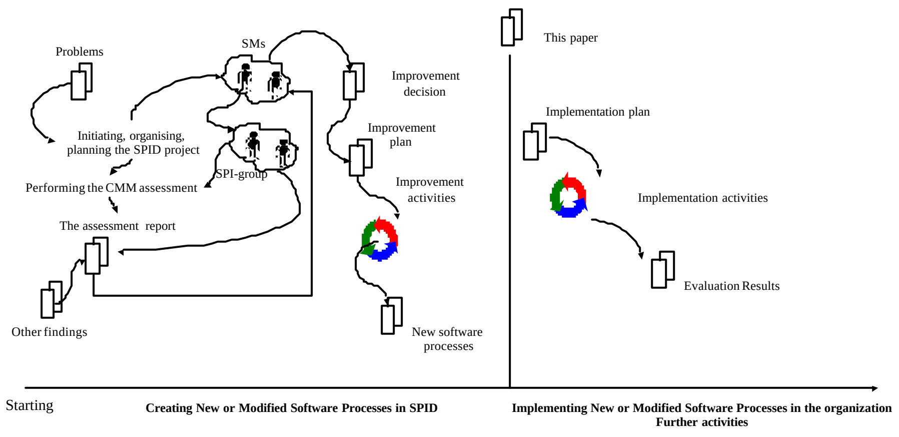
This research started before the merger between the two companies in an IS organisation called Clinical Research and Information Management (CRIM) at the former Astra Hässle in Sweden and continued later in the new IS organisation, which then changed its name to Development IS (DevIS). DevIS supports clinical and pharmaceutical projects, Regulatory Affairs and Product Strategy and Licenses at AstraZeneca R&D Mölndal. DevIS is also responsible for influencing the development of the global clinical research processes and IS/IT tools in AstraZeneca. DevIS comprises 90 people including contractors, most of whom have backgrounds in IS/IT.

Many regulatory authorities require that pharmaceutical companies and their software organisations comply with GXP (Good *Manufacturing* Practice, Good *Clinical* Practice, and Good *Laboratory* Practice) rules. GXP rules are the authorities' quality requirements to pharmaceutical companies for ensuring patient health, the quality of processes (e.g. clinical studies or software development) and the quality of products (e.g. tablets or software). As a software organisation in the pharmaceutical business, DevIS must address many quality requirements. One fundamental requirement is that DevIS must be able to show the authorities, by documented evidence, that software development activities (e.g. software change control, software validation, and data processing and storage) are being performed in compliance with quality requirements. Therefore every software project regulated by GXP requirements should carefully apply all quality rules and be able to show by documented evidence that the software is compliant with the related GXP requirements. The company long ago adopted standard operation procedures that explicitly describe the company's software quality rules. These standard operation procedures should be applied in all information systems regulated by GXP requirements.

Employees of DevIS are basically engaged with software development, software maintenance and software operation activities. The software development activities occur in two forms: 1) development of totally new software products (software development) and 2) developing or changing existing software products (software maintenance). A typical software development project at DevIS is scheduled to take between six months and one year and includes analysis, design, construction, testing, and validation. Software maintenance activities can consist of changes in the code or developing a completely new application for existing software products. Software products in DevIS include the software and all related documentation (e.g. user requirement specification, test plan, validation plan, validation report, user manuals etc.).

3.2 The Problem Area

The results of a problem analysis performed in early 1999 in one software development group within DevIS showed a need for improving software project disciplines and providing guidelines to understand the standard operation procedures and GXP rules. The director of DevIS initiated an improvement project called Software Process Improvement at DevIS (SPID) to understand the existing problems and improve the organisation's software processes. The following figure illustrates a rich picture of the SPID project.



Problems
SMs
Improvement decision
This paper
Implementation plan
Initiating, organising, planning the SPID project
Improvement plan
SPI-group
Performing the CMM assessment
Improvement activities
Implementation activities
The assessment report
Evaluation Results
Other findings
New software processes
Starting
Creating New or Modified Software Processes in SPID
Implementing New or Modified Software Processes in the organization
Further activities

Figure 3. The rich picture of SPID (see Checkland and Scholes 1990)

The SPID project was initiated, organised, planned, and performed during the period of April 1999 to May 2000 and aimed to improve DevIS' software processes. A maturity assessment using a modified CMM-based (Capability Maturity Model) assessment method, QBA (see Arent and Iversen 1996), showed that DevIS was by then a level one organisation and addressed improvement possibilities in all analysed KPAs (Key Practice Areas). An improvement report based on the assessment's findings and other findings from earlier improvement initiatives at DevIS addressed six improvement activities. The steering committee of SPID gave priority to the following improvement activities (improvement decision) from the improvement report:

1. To establish a minimum documentation level for documenting the results of software projects and create the software documentation process.
2. To improve processes for software validation, software change management, and document version control.
3. To create a template library including templates for documentation of software development activities, such as: user requirement specification, design specification, test plan, and validation plan.

The SPI group including (5 software engineers, 2 assisting consultants, and the author) started planning and performing improvement activities over a four-month period. This initial phase of SPID was scheduled to be finished in June 2000. The implementation activities for implementing the newly created software processes in the whole organisation start in August 2000.

4 DISCUSSION

This section discusses SPID on the basis of knowledge creation framework described earlier in this paper. We present a map illustrating the knowledge creation process in SPID including the software process improvement steps, the knowledge creation processes, the knowledge created, and the actors and the organisational levels involved.

During the initiating phase, we realised that the standard CMM maturity assessment developed by the SEI (see Zubrow *et. al.* 1994) was too general for our situation. We adapted a simplified CMM-based assessment method, QBA (see Arent and Iversen 1996), that focuses on level two and modified it to fit DevIS's terminology and needs. The assessment indicated that we needed to focus more on software validation, change, and version control, and on creating templates rather than on processes related to project management (e.g. software project planning, software project tracking and oversight). The software organisation aimed to create only a few processes based on practitioners' experiences and the organisation's quality requirements. The SPI-group working with improvement activities spent a grate deal of time studying and understanding the CMM, the organisation's quality requirements, the organisation's existing standards, and the software engineering literature to create its own understanding of what is needed to create the new processes. However, we succeeded by following the IDEAL model in organising and planning for the performance of the improvement activities (the I, D, E and partly A, and L steps in IDEAL) in a systematic way. This allowed us to see very early in the project the starting point, the finishing point and all activities included in reaching our targets. We followed IDEAL's main steps very naturally and learned how to systematically go about creating the new or modified software processes. However, the path from knowing what to do to doing it in practice was neither easy nor clear.

4.1 The knowledge creation process in SPID

According to the organisational knowledge creation framework presented earlier in this paper, the organisational knowledge creation process starts at the socialisation phase at the group level and continues to the externalisation phase, aiming to create new explicit knowledge based on the interaction between tacit and explicit knowledge. The created explicit knowledge will, then be combined with other already existing explicit knowledge in the organisation. The new or modified knowledge is subsequently put into practice through learning by doing to become tacit knowledge. In our software process improvement project the first organisational knowledge creation process started in the socialisation phase creating sympathised knowledge about the SPI concept through interaction between tacit to tacit knowledge. The author arranged meetings with the management to introduce the concept of SPI (creating SPI-knowledge). These meetings were held on an individual level between the director of the software organisation, other software managers, and the author aiming to learn about the SPI and the possibilities for gaining benefits by doing an SPI project. Other meetings with the same contents were held with the SPI-group for similar purposes. The created tacit SPI-knowledge then became explicit SPI-knowledge through a dialogue in the externalisation phase in which conceptualised knowledge was created mostly by the author, the software managers, and the assisting consultants. The author arranged other meetings to identify the initial improvement infrastructure needed to carry out the project. The created SPI-knowledge in this phase was mostly in the form of project specifications including information about the SPI plan, resources needed, role descriptions, goals, and responsibilities. The created explicit SPI-knowledge (the project specification) was then combined with other existing knowledge and experience in improving software processes, and other improvement models at the software organisation to create systematic knowledge. In this phase (combination), knowledge was created mostly at the individual and group levels through a dialogue between all actors. The operational SPI-knowledge was then created through practising software process improvement activities. The author, the software engineers, the software managers, and the assisting consultants were all involved in making SPI happen in the organisation. In this phase the created explicit SPI-knowledge became tacit SPI-knowledge through practice. This means that, in our software process improvement project, all organisational knowledge creation phases involved all actors in individual and almost group levels in the creation of SPI-knowledge.

The other organisational knowledge creation process in our project deals with creating SE-knowledge (Software Engineering Knowledge). This process also started in the socialisation phase through externalisation, and combination to create SE-knowledge. A modified CMM assessment done in three software projects involving three software project managers and two software developers to identify the current level of software process problems. The results and the recommendations of the assessment were identified and documented. This action led to the creation of the first explicit SE-knowledge about the current maturity level of the organisation based on the results of the assessment. The author and the assisting consultants were involved in creating this knowledge. This knowledge was then combined and integrated with other organisational requirements, and findings from earlier improvement activities. The author interpreted the authorities' and organisation's quality requirements on software development practice and created explicit knowledge about the quality requirements in an individual level. The created knowledge about the quality requirements, earlier improvement findings, and assessment findings were summarised on an improvement suggestion report created by the author and the assisting consultants and presented to management for a decision. The created explicit SE-knowledge was then transferred to group levels when the author held presentations to give results of the assessment in different meetings for different groups. This explicit SE-knowledge was then used by the SPI-group as input to create the minimum baseline level for documentation and other new software processes.

The next step involved the process of capturing practitioners' ideas about software practices and forming them into explicit knowledge. The author arranged meetings in which the software engineers and project managers shared their experiences and ideas about the software processes (in the socialisation phase). We focused in this phase on creating a common understanding of the software process problems and get agreeing on the ideal software processes. In the next phase (externalisation) we focused on creating explicit knowledge about the new software processes based on the practitioners' ideas. We created some drafts for the new processes and discussed and changed the contents of the processes several times until we could agree upon an acceptable level. This explicit SE-knowledge was created by the SPI-group at the individual and group levels. The software managers were not involved in creating SE-knowledge. This phase was one of the most important and also one of the most difficult phases in our SPI project. This suggests that such a project should not create new processes and standards detached from practices. New or modified processes should be created based on practitioners' experiences from practice. This suggestion is supported in several studies that indicate that routines and standards are learned and developed in practice, not from formal explicit standards procedures (Brown and Duguid 1991, Arent and Nørbjerg 2000).

The following tables illustrate the knowledge creation process in our project: *: Total, ~: Partly, -: Nothing.
SPID: Software Process Improvement at Development IS (our SPI project)
SPIer: Software Process Improver, (the author)
SMs: Software Managers
SPICs: Software Process Improvement Consultants: (the assisting consultants)
SEs: Software Engineers
OKCL: Organisational Knowledge Creation Level
Ind.: Individual, Gro.: Group, Org.: Organisational
OKCP: Organisational Knowledge Creation Process
Soc: Socialisation, Ext.: Externalisation, Comb.: Combination, Intern.: Internalisation

SPI Phases	SPI Activities	Organizational Knowledge Creation Processes SPI-Knowledge & SE-Knowledge	OKCP Soc.Ext.Comb.Intern.	OKCL Ind.Gro. Org	Roles SPIerSMsSPICsSEs
Initiating	• Establishing the SPI project. * • Establishing the initial improvement infrastructure. * • Defining the roles and responsibilities for the infrastructure. * • Assigning the initial resources for performing the SPI project. * • Creating an SPI plan. * • Defining the goals of the SPI project. * • Establishing the steering-, and SPI-groups. *	The SPIer arranged meetings with management for introducing the concept of SPI (SPI-knowledge). These meetings held on an individual level between the director of the software organization and other SMs and theSPIer and were aimed at learning about the SPI and the possibilities for gaining organizational benefits by doing an SPI project. Similar other meetings with same contents were held with the SPI-group for similar purposes. **SPI-knowledge was created** .	* * ~ *	* ~ -	* * ~ ~
		The SPIer arranged other meetings to identify the initial improvement infrastructure needed for performing the SPID project. These meetings were held both with the director of the software organization and with the SPIer's supervisor, and other researchers working with SPI. **SPI-knowledge was created** .	* * ~ *	* ~ -	* * * -
		The researcher created a project specification including information about the SPI plan, resources needed, role descriptions, goals, and responsibilities. The management established a steering committee and a reference group for the SPID. **The SPID project was established. The created knowledge in this phase was dominated by SPI-knowledge.**	* * ~ *	* ~ ~	* * * -
Diagnosing	• Initiating the SPI action plan. ~ • Doing appraisal activities (assessment). * • Identifying results and recommendations from appraisal activities (suggested improvement measurement). * • Discussing the identified results and recommendations from appraisal activities with the SMs. *	The SPIer in collaboration with an SPI consultant and a member of the SPI-group worked for two months to establish a modified CMM assessment survey. An SPI action plan was initiated.**Explicit SPI-knowledge was created**.	* * * *	* * ~	* ~ * *
		The CMM assessment was done on three software projects. The results and the recommendations were identified and documented. Knowledge about the software problems and the improvements was created based on practitioners' earlier experiences of software development on an individual level. **The improvement report was written . Explicit SE-knowledge was created.**	* * * *	* * ~	* - * *
		The explicit SE-knowledge created based on the results and the recommendations from appraisal activities were networked with the steering committee and also partly to the software organization. **The created knowledge in this phase was dominated by SE-knowledge.**	* * * *	* * ~	* * ~ -
Establishing	• Giving priority to the issues that the organization has decided to address with its improvement activities. * • Developing strategies for pursuing the improvement suggestions. * • Completing the SPI action plan draft. * • Developing measurable goals. * • Developing the final version of the SPI action plan. * • Defining metrics necessary to monitor progress. - • Making commitments for resources and training provided for the SPI-group. *	The issues that the steering committee decided to address with its improvement activities were prioritized. The SPIer developed strategies for pursuing the improvement suggestions and further developed an action plan, and measurable goals, which were discussed with the steering committee. **An action plan was created. Explicit SPI-knowledge was created.**	* * ~ *	* ~ -	* * - -
		The management created a technical working group (the SPI-group). The SPIer made commitments for resources and training provided for the SPI-group. The created knowledge about the software problems, improvement suggestions, and action plan were networked between the steering committee, and the SPI-group. **Explicit SPI-knowledge was created. The created knowledge in this phase was dominated by SPI-knowledge**	* * * *	* * -	* ~ - *
Acting	• Planning and performing the suggested improvement measures.* • Developing plans to execute pilots to test improvement measures. - • Perform appraisal activities (testing new processes) to evaluate the improvement measures. -	The SPI-group created explicit SE-knowledge about the new and modified software processes. The suggested improvement measures were planned and performed. **The new and modified processes were created. New explicit SE-knowledge was created** .	* * * *	* * -	* - * *
Learning	• Learning and documenting the lessons. * • Collecting the metrics on performance and goal achievement. ~ • Establishing a project database that can be used as a source of information for personnel involved in the next pass through the IDEAL model for further software process improvement initiatives. * • Proposal continues research activities. *	The SPIer documents the lessons learned from the SPID. The documented knowledge will be transferred to other software development groups. Practitioners will get trainees in how to use the new and modified software processes and encourage learning these processes by practicing them in their daily work. **The created explicit SE-knowledge will become tacit SE-knowledge in the whole organization.**	* * * *	* * *	* * * *
		The SPIer arranged other meetings to identify the initial improvement infrastructure needed for implementing the new processes. These meetings were held with the director of the software organization and with the SPIer's supervisor, and other researchers working with SPI.	* ~ ~ *	* ~ -	* * * -

Figure 4. The organisational knowledge creation process in SPID

5 LESSONS LEARNED

We have analysed a software process improvement project from an organisational knowledge creation perspective using a framework based on Nonaka and Takeuchi (1995) of organisational knowledge creation process. From the perspective of this framework we believe that it is useful to view such a project as an organisational knowledge creation process. On the basis of our experiences from this study we believe that certain types of created knowledge within an SPI project deal with the fundamentals of Software Process Improvement (SPI-knowledge) such as: management of SPI initiatives, issues related to how such an initiative should be guided, and issues related to SPI's focus on target(s) (see Aaen *et al.* 2001). Others deal with issues related to Software Engineering practice (SE-knowledge) such as: project planning, quality assurance, change control, and configuration management (see Pressman 1997).

To illustrate the knowledge creation process we have applied the Nonaka and Takeuchi's framework to one SPI project and created a map illustrating: the software process improvement phases, the activities, the created knowledge, the actors involved, and the organisational levels on which knowledge is created. On the basis of our experiences from this study we suggest a number of lessons relevant for future software process improvement projects, the SPI practice, and the effect of organisational knowledge creation on SPI.

Lesson one : *Two related knowledge domains (Software Process Improvement (SPI)-, and Software Engineering (SE)-knowledge) are involved in the knowledge creation process in an SPI project.* Two types of knowledge, i.e. SPI-, and SE-knowledge, play an essential role in SPI activities and the knowledge creation practice involved in an SPI project. The knowledge creation behaviour is completely different in these two knowledge domains.

Lesson two: **a:** *The project manager of the SPI project, software managers, and assisting consultants are the key actors involved in the creation of SPI-knowledge.* The most involved actors in creating SPI-knowledge during the very initial phase of the project are the SPI project manager, and the software managers. The SPI-knowledge is created mostly during the initiating, establishing, and learning phases. The SPI project manager is most involved in creating SPI-knowledge. The assisting consultants and the software managers contribute to combining the SPI-knowledge with their ideas and experience. Later on in the project other actors will become involved in the knowledge creation process.

Lesson two: b: *The software engineers and the SPI project manager are the key actors involved in the creation of SE-knowledge.* The SE-knowledge is created primarily during the diagnosing, acting, and learning phases. During the diagnosing phase the SPI project manager, the software engineers, and the assisting consultant are involved. Later on in the project only the SPI project manager and the software engineers are involved in the creation of SE-knowledge. SE-knowledge is created on the basis of software engineers' experience and ideas about software processes. The software managers are not involved in any phases of the project for creating SE-knowledge.

Lesson three: a: *The knowledge creation process in the case of SPI-knowledge happens mostly on the individual level and sometimes on the group level.* The very first SPI-knowledge is created on an individual level through interaction between the SPI project manager, and the assisting consultants. This knowledge is then transferred to the group level in which the software managers are involved.

Lesson three: b: *The knowledge creation process in the case of SE-knowledge happens mostly on the group level and sometimes on the organisational level.* SE-knowledge about current software process problems and new software processes is created on the basis of software engineers' ideas on a group level. This knowledge is then transferred to other organisational levels.

These lessons agree with Nonaka and Takeuchi's suggestion of starting the organisational knowledge creation process at the socialisation phase on the team level. However, in our SPI project, creating SPI-knowledge started in the socialisation phase on the individual level and then passes through almost all other organisational knowledge creation phases to the group level. Creating SE-knowledge in our project also started in the socialisation phase mostly on the group level and then passes through almost all other organisational knowledge creation phases to the organisational level. If an SPI project ends before an implementation of the new software processes in the whole organisation (as it did for us) then the internalisation of the SE-knowledge might happen only during a test period in a pilot project (if any) in the improvement phase. The organisational knowledge creation process for SE-knowledge passes to the organisational level first when the created software processes are implemented in the whole organisation to become a part of all practitioners' daily work.

The implementation of newly created software processes is an issue of implementing *changes* in practitioner's daily work. These changes should be organised, planned, and implemented in the whole organisation to be a part of the organisation's daily practice. As mentioned before, one great challenge for DevIS is to find a way to implement the new or modified software processes in the organisation. An important question for further research is how can theories such as organisational learning and change management support, understand, and enhance implementation initiatives of new processes at DevIS?

ACKNOWLEDGEMENT

This research was sponsored by AstraZeneca R&D Mölndal in Sweden in collaboration with the Viktoria Institute and the Institute of Computer Science in Aalborg, Denmark.

REFERENCES

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement. *Scandinavian Journal of Information Systems,* Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 123-146.

Ami (1992). *Ami, application of metrics in industry, A Qualitative Approach to Software Management.* CSSE South Bank University, London, pp. 1-35.

Arent, J. and Iversen, J. (1996). *Development of a Method for Maturity Assessments in Software Organizations based on the Capability Maturity Model.* Aalborg University: Department of Computer Science.

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science.*

Baskerville, R. and Wood-Harper, A. T. (1996). A Critical Perspective on Action Research as a Method for Information Systems Research. *Journal of information technology,* Vol. 11.

Brown, J. S. and Duguid, P. (1991). Organisational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning and Innovation. *Organization Science,* Vol. 2, No. 1, pp. 40-57.

Checkland, P. and Scholes, J. (1990). *Soft System Methodology in Action.* John Wiley & Sons.

Debou, C. (1997). SPI Success Factors: Toward More Business Orientation. *Software Process Newsletter, IEEE Computer Society,* No. 10, pp. 15-18.

Firebaugh, M. W. (1989). *Artificial Intelligence – A Knowledge-Based Approach.* PWS-KENT Publishing Company.

Goldenson, D. R. and Herbsleb, J. D. (1995). *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-009.

Halloran, P. (1999). Organizational Learning from the Perspective of a Software Process Assessment and Improvement Program. *Proceedings of Hawaii International Conference on Systems Science,* Los Alamitos, CA: IEEE Computer Society Press.

Hayes, W. and Zubrow, D. (1995). *Moving On Up: Data and Experience Doing CMM-Based Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-008.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. C. (1997). Software Quality and the Capability Maturity Model. *Communications of the ACM,* Vol. 40, No. 6, pp. 30-40.

Humphrey, W. (1989). *Managing the Software Processes.* Addison-Wesley.

Johansen, J. and Mathiassen, L. (1998). Lessons learned in a National SPI Effort. EuroSPI 98 Gothenburg, Sweden, November 16-18, pp. 5.1-17.

Kuvaja, P. and Bicego, A. (1994). BOOTSTRAP - a European assessment methodology. *Software Quality Journal,* Vol. 3, pp. 117-127.

Larsen, E. Å. and Kautz, K. (1997). Quality Assurance and Software Process Improvement in Norway. *Software Process - Improvement and Practice*, Vol. 3, No. 2, pp. 71-86.

Mashiko, Y. and Basiili, V. R. (1997). Using the GQM Paradigm to Investigate Influential Factors for Software Process Improvement. *Journal of Systems and Software*, Vol. 36, pp. 17-32.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

McFeeley, B. (1996). *IDEAL. A User's Guide for Software Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Handbook CMU/SEI-96-HB-001.

Nonaka, I. and Konno, N. (1998). The Concept of "Ba": Building a Foundation for Knowledge Creation. *California Management Review,* Vol. 40, No.3, pp. 40-54.

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

Paulk, M., Curtis, B., Chrissis, M. and Weber, C. (1993). *Capability Maturity Model for Software Version 1.1.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-96-TR-24.

Polanyi, M. (1996). *The Tacit Dimension. Doubleday*. Garden City.

Pressman, R. S. (1997). *Software Engineering, a practitioner's approach*. Fourth edition, McGraw-Hill, International Editions, Software Engineering Series.

Stelzer, D., Mellis, W. and Herzwurm, G. (1998). Technology Diffusion in Software Development Processes: The contribution of Organizational Learning to Software Process Improvement. In Larsen, T. and McGuire, E. (Eds.): *Information Systems Innovation and Diffusion: Issues and Directions*, Idea Group Publisher.

TickIT (1995). *TickIT, Guide to Software Quality Management System Construction and Certification using ISO 9001:1994.* Issue 1.0, DISC TickIT Office.

Thomson, H. E. and Mayhew, P. (1997). Approaches to Software Process Improvement. *Software Process-Improvement and Practice*, Vol. 3, No. 1, pp. 3-17.

Wohlwend, H. and Rosenbaum, S. (1994). Schlumberger's Software Improvement Program. *IEEE Transactions on software engineering,* Vol. 20, No. 11, pp. 833-839.

Zubrow, D., Hayes, W., Siegel, J. and Goldenson, D. (1994). *Maturity Questionnaire*. Special Report CMU/SEI-94-SR-7, Software Engineering Institute.

ANALYSING AN SPI PROJECT WITH THE MAP FRAMEWORK

Pouya Pourkomeylian

Abstract: Software Process Improvement (SPI) is a recognised systematic approach for improving the capability of software organisations. Initiatives of this kind have met a number of difficulties such as: scaling the SPI initiatives, setting realistic goals, coping with the complexity of organisational changes, and dealing with the organisational culture. Organisations with no previous experience of SPI might therefore run the risk of the first initiative being the last. This paper gives the results of a collaborative research project in which the first SPI initiative in an organisation was analysed according to a framework that maps the characteristic features of SPI, (the MAP framework). On the basis of our findings it is argued that the first SPI initiative: 1) should be organised as a project aiming to improve a few software processes, 2) should satisfy organisational goals rather than routinely follow a normative model for reaching a maturity level, and 3) should include a MAP analysis early in the project to better understamd the nature of SPI activities.

Keywords: Software Process Improvement (SPI)

1 INTRODUCTION

Software Process Improvement (SPI) is a systematic approach to improve software processes in organisations. The approach was developed by the Software Engineering Institute (SEI), inspired by the work of Watts Humphrey (1989). The basic idea of SPI is to focus on software processes as social institutions with a complex interplay of people, methods, tools, and products (Aaen *et al.* 2001).
An SPI initiative is cyclic in nature and includes different phases 1) Initiating, 2) Diagnosing, 3) Establishing, 4) Acting and 5) Learning as expressed in the IDEAL model (McFeeley 1996). In the initiating phase preparations are made to carry out the SPI effort. It includes plans, schedules, and infrastructure. The next step is devoted to diagnosing the current maturity level of the organisation's software processes. This information will become the basis for focused improvement projects in the next step. Each project creates new or enhanced software processes, which are verified and eventually implemented in the whole organisation to improve the software engineering practices. The final phase is focused on continued improvement, including measurements of the newly created software processes and documenting lessons learned from the SPI efforts (McFeeley 1996, Zahran 1998).

Many organisations have been inspired by the concept of SPI and started SPI initiatives. Achieving success with SPI has however proven to be a difficult challenge. Many organisations do not succeed in their improvement activities and others have problems with the implementation of new processes in the organisation (Tryde *et al.* 2001). Different factors such as scaling the SPI initiative, setting realistic goals, coping with the complexity of organisational changes, and dealing with the organisational culture have made it hard to achieve success in SPI initiatives (Goldenson and Herbsleb 1995, Herbsleb *et al.* 1997, Mashiko and Basiili 1997, Johansen and Mathiassen 1998).

For an organisation with no earlier experience in SPI - a novice organisation - the first initiative may consequently run the risk of being the last.

According to Aaen *et al. (*2001), organisations that start SPI efforts should consult and find inspiration and guidance in the literature. They argue that these organisations can avoid the pitfalls that have led to failure in other organisations by learning from successful initiatives. However, following this advice is not easy: the SPI literature is extensive and it is growing and there are no authoritative sources outlining the underlying rationale for SPI.

A large body of knowledge about SPI has become available in recent years, including specific models (Paulk *et al.* 1993, Kuvaja 1994), concepts to support practical use of the models (McFeeley 1996, Zahran 1998), experience reports (Goldenson and Herbsleb 1995, Johansen and Mathiassen 1998), and critical evaluations (Curtis 1994). A survey of SPI literature and a list of the key ideas in SPI are presented by Aaen *et al.* (2001). They provide a conceptual map that describes three fundamental aspects of SPI defined through nine elementary ideas. According to the authors, SPI is based on these ideas, which offer specific answers to specific concerns. SPI has three fundamental concerns: the *management* of SPI, the *approach* taken to guide the SPI initiatives, and the *perspective* used to focus attention on the SPI goals, thus the MAP. It addresses among others the following crucial questions: (1) What are the characteristic features of SPI initiatives? And (2) What are the key benefits and risks related to SPI initiatives?

According to Aaen *et al*. (2001), the management of SPI initiatives builds on three ideas: 1) the SPI activities are *organised* as dedicated efforts, 2) all improvement efforts are carefully *planned,* and 3) *feedback* on effects on software engineering practices is ensured. The approach to SPI initiatives is guided by three additional ideas: 1) SPI is *evolutionary* in nature, 2) SPI is based on idealised, *normative* models of software engineering, and 3) SPI is based on a careful creation and development of *commitment* between the actors involved. Finally, the perspective on the SPI target is dominated by three ideas: 1) SPI is focused on software *processes*, 2) the practitioners' *competencies* are seen as the key resources, and 3) SPI aims at changing the *context* of the software operation to create sustainable support for the actors involved. The MAP defines the objectives, but also points to a number of pitfalls for each idea.

Using the MAP, this study analyses an ongoing SPI project undertaken as the first SPI initiative in an organisation with the aim of identifying key success factors in SPI initiation in a novice organisation. On the basis of his experiences the author will argue that such an analysis may help other novice organisations in finding ways to increase their chances of successfully planning, organising, and running SPI activities and of minimising the risks of failure. This study tries to find answers to the following question:

What are the key success factors for a first SPI effort?

The next section briefly discusses the research approach. Section 3 introduces the case and section 4 presents the results of the analysis of the SPI initiative and discusses the findings with regard to the research question stated above. Section 5 concludes the paper by presenting the key success factors in the SPI effort in the organisation.

2 THE RESEARCH APPROACH

The research presented here covers an ongoing SPI effort that started in April 1999 and whose first full cycle is planned to end in June 2001. So far in this period, the SPI effort as a whole has been established, a CMM based assessment has been performed, the first three software processes have been locally piloted, and an implementation plan to put these and further new processes into practice in the whole organisation has been developed. At the time of writing, in March 2001, implementation activities are in full operation aiming at rolling out the newly created software processes in the entire organisation (see Pourkomeylian 2001c).

The author has been the project manager and the main driving force behind the SPI endeavour and has actively participated in the activities to initiate, organise, plan, and conduct the initiative during the two-year period. In this study he reflects on the SPI initiative and tries to provide lessons useful for understanding the field of SPI in general and in practice. The evaluation of this case is based on the author's subjective observations of the SPI effort, his experience as project manager, and informal discussions between the author and employees participating in the SPI effort both during the effort and after performing certain improvement activities.

This paper is one of the results of a collaborative practice research project (see Mathiassen 2000) between a research institution and the software development organisation at AstraZeneca Mölndal in Sweden. The basic approach in collaborative practice research is action research, but more traditional practice studies and experiments are applied to serve specific needs. This paper can be viewed as a case study in which a theoretical framework, the MAP (see Aaen *et al.* 2001), is used to analyse the SPI initiative and to provide guidance for success in planning, organising, and carrying out SPI activities.

Mapping the SPI effort

In August 2000 an SPI expert and the author analysed the SPI project in the software organisation. For this purpose we used the MAP framework. The goal was to identify the concrete elements of the SPI effort with regard to the MAP framework and whether it had missed any crucial SPI features. For each fundamental concept and its accompanying ideas, the SPI expert and the author determined whether and to what extent these ideas had been applied and followed in concrete situations during the course of the SPI project. We elucidated the reasoning behind utilising or not utilising each specific SPI idea and evaluated the effects that pursuing or not pursuing an idea had had on the SPI effort. We also identified actual pitfalls for every SPI idea. The author completed the MAP analysis in February 2001 for improvement activities carried out after August 2000. Finally, we implicitly assessed the usefulness of the MAP for understanding and implementing SPI initiatives.

3 THE CASE

This study was conducted at AstraZeneca, one of the world's leading pharmaceutical companies. AstraZeneca is a research-driven organisation with a large range of medical products designed to fight diseases. The company employs over 25,000 people, some 10,000 as R&D personnel and 12,000 people in production in 20 countries. It has an extensive global sales and marketing network and had a R&D investment in 1999 of about US $2 billion.

The study was performed within an IS organisation called Development IS in AstraZeneca R&D Mölndal in Sweden. DevIS supports clinical and pharmaceutical projects, regulatory affairs and product strategy and licenses at AstraZeneca R&D Mölndal. DevIS is also responsible for the development of the global clinical research processes and IS/IT tools in AstraZeneca. DevIS comprises 110 people including contractors, most of whom have backgrounds in IS/IT. DevIS employees are basically involved in software development, software maintenance, and software operation activities. The software development activities occur in two forms: 1) development of totally new software products - original software development - and 2) further development, change or adaptation of existing software products - software maintenance. A typical DevIS software development project is scheduled to take between six months and one year and includes analysis, design, construction, testing, and validation. DevIS software products include software and all related documentation, e.g. user requirement specification, test plan, validation plan, validation report, user manuals etc..

Many regulatory authorities require that pharmaceutical companies and their software organisations comply with GXP (Good *Manufacturing* Practice, Good *Clinical* Practice, and Good *Laboratory* Practice) rules. GXP rules are the authorities' quality requirements to pharmaceutical companies for ensuring patient health, the quality of processes (e.g. clinical studies or software development) and the quality of products (e.g. tablets or software). As a software organisation in the pharmaceutical business, DevIS must address many quality requirements. One fundamental requirement is that DevIS must be able to show the authorities, by documented evidence, that software development activities (e.g. software change control, software validation, and data processing and storage) are being performed in compliance with quality requirements. Therefore every software project regulated by GXP requirements should carefully apply all quality rules and be able to show by documented evidence that the software is compliant with the related GXP requirements. The company long ago adopted standard operation procedures that explicitly describe the company's software quality rules. These standard operation procedures should be applied for all information systems regulated by GXP requirements.

The Problem Area

An informal problem analysis made in early 1999 in one of DevIS's software development units showed that DevIS's software project practice needed improvement. There was also a need of providing guidelines to understand the standard operation procedures and GXP rules. Many practitioners working in different software projects pointed to this subject for improvement by sending email to an analysis group responsible for the gathering software professionals' ideas for improvement. Management at that time did not have detailed knowledge of the depth of the problem and how to improve the software project practice. The director of DevIS thus initiated a project to analyse and understand the problems of software project practice and, if possible, to improve the software project discipline.

Early in 1999 the author worked as Quality Manager of DevIS, responsible for quality issues. At that time we did not know how and where to start improvement efforts. However, the author had heard about successful results from other organisations using SPI and the CMM (Capability Maturity Model) for improving the capabilities of software organisations. After further study of the SPI literature the need of using two approaches for the improvement activities became clearer: 1) a structured and systematic model for planning organising, analysing, and improving software practices, 2) a model for focusing specifically on software project problems. After meetings with the director of DevIS and an SPI expert discussing different approaches to improvement activities we decided to start a SPI project, later called SPID (Software Process Improvement at DeveIS), using the IDEAL model for planning, organising, and running SPI activities, and using the CMM to focus on level 2 key process areas (see McFeeley 1996, Goldenson and Herbsleb 1995, Hayes and Zubrow 1995).

The first step was to establish the SPI project's organisation. The SPID organisation was established in April 1999 and included: one project manager - the author, responsible for planning and running the project; a steering committee headed by the director of DevIS and three software managers responsible for dedicating resources to the project, making decisions about the project's focus and approving the results; a reference group including three project managers and two software developers responsible for improving the software processes; and an SPI working group including two SPI consultants and the author, responsible for documenting the SPI project.

The second step in SPID was to diagnose the current maturity level of DevIS's software projects. A maturity assessment was performed in May 1999 using a modified CMM-based assessment method, QBA (Arent and Iversen 1996), which has a focus on the CMM level 2 key process areas. An external SPI consultant helped us to conduct the assessment and to summarise the results. Three completed software projects were the focus of the assessment (see for more detail, Pourkomeylian 2001a), which was especially concerned with project and requirements management and addressed improvement possibilities in all key process areas analysed - requirements management, project planning, project tracking and oversight, quality assurance, and configuration management. Software subcontract management which is also a CMM level 2 key process area was out of the focus of the assessment because this key process area was not widely used in DevIS's software project.

On the basis of the results of the assessment, the author produced an improvement report in October 1999 addressing six improvement activities, with a specific focus on software documentation and software validation to satisfy a most important demand, namely that the pharmaceutical industry has to document all software engineering activities to comply with health regulations. In the case of inspections the company must be able to show documented evidence that a specific task, e.g. that implementation of change in a software product has been performed in accordance with predefined standard procedures.

The steering committee of SPID decided to give priority to software documentation and software validation processes through:

- Creation of a template library including templates for the documentation of software development activities such as: user requirement specification, design specification, test plan, and validation plan.
- Creation of a software documentation process including a minimum documentation level for documenting the results of software projects.
- Defining processes for software validation, software change management, and document version control.

As the next step an improvement plan was created by the author and the SPI working group started to work on improvement activities. This group planned and performed improvement activities over a period of four months, which resulted in the creation of new software process guidelines - a software documentation guideline, a software validation guideline, a software change control and document version control guideline - and developed the template library. The newly created software processes were presented to the steering committee and further modified based on the committee's feedback on improvements.

An implementation plan was created to address the implementation activities necessary to initiate the new processes in the whole organisation. The plan was developed through a Participatory Implementation Workshop (PIW) (Andersson and Nilsson 2000). Two external consultants were invited to hold a PIW workshop at the software organisation to help us identify the most important factors needed for creating an implementation strategy for the newly created software processes. The PIW workshop was held in May 2000 at AstraZeneca. The author, one SPI consultant, one project manager, all involved in SPID, and the two external PIW consultants participated in the workshop (see for more detail, Pourkomeylian 2001c). The implementation plan was then presented to the steering committee. The steering committee of the project accepted the refined software processes and the implementation plan and decided to implement the processes throughout DevIS.

The acceptance of the software processes and decision to implement them throughout the whole organisation were a first success of the initiative. The implementation activities have been scheduled to take place between August 2000 and June 2001. One important aim was to change the context in which the new software processes should operate in. Therefore, among others a trainee program was scheduled for all practitioners at DevIS. The implementation phase also includes further improvement activities in which the processes will be enhanced on the basis of experience of using them in practice. This phase will result in a new version of the software process guidelines in June 2001.

Now, at the time of writing in March 2001, the SPID is in the implementation phase adapting the newly created software processes in every software project and continuously improving the processes and templates based on experience from practical use. An SPI unit has been established at the company level at AstraZeneca in Mölndal to support practitioners when they apply the new processes and use the templates. Thus, the project has reached far with respect to gaining management and practitioners' commitment. The processes and templates are being used in 12 software projects and are appreciated by practitioners working in these software projects. The SPI unit measures and documents the feedback and further improvement suggestions from the practical use of the new processes and templates in these software projects.

4 THE FINDINGS

It was however a long way and some of the difficulties might have been avoided if the organisation would have been prepared. Thus, this section discusses the most characteristic features of SPI that affected the SPID project on the basis of the MAP framework (see Aaen *et al.* 2001). Table 1 summarises the concrete SPID activities as evaluated from a MAP perspective.

Concern	Idea	Aspiration/ Objective	Followed	Reason	Concrete situation in the SPID project	Evaluation of the Situation/ Created effect	Experienced problems
Management of SPI	*Organisation*	Create a dedicated effort adapted to the conditions of the organisation.	Yes	To create a dedicated effort adjusted to the DevIS conditions.	SPID is organised as a project like all other software projects at DevIS having a specific budget and resources. The SPID organisation consists of a reference group, a steering committee, a working group, and a project manager.	Achieved management commitment. Achieved resource dedication.	Inadequate resources. Difficult co-ordination. Weak emphasis.
	Plan	Plan goals, activities, responsibilities, and co-ordination.	Yes	To co-ordinate the project and to know: Where to go, who does what and when, and what are the results.	An SPI plan has been created to identify the milestones, deliverables, responsibilities, and activities needed in the project. The plan is a framework for action rather than a procedure to follow.	Established clear roles, responsibilities, and deliverables Provision of sufficient budget and resources for the project.	Feeling of some stress with regard to deadlines.
	Feedback	Measure and assess benefits.	No	To establish a special reference group that consists of experienced project managers and software developer who can give feedback on the improvements as we go along.	The SPID hasn't defined in detail how feedback on improvements should be measured.	Spend time on other aspects. Uncertain whether the project is on the right track.	Impossibility to measure and document quality and progress of the project.

Approach to SPI	*Evolution*	Learn by experience and employ stepwise improvements.	Yes	To learn SPI by doing it. To adapt the concept of SPI to the organisation to apply stepwise improvements.	SPID aims at learning from the SPI concept and at improving a few software processes in an evolutionary way by: Identifying the current software process problems and identifying the minimum documentation requirements SPID has created processes for change management and software validation, documentation and templates.	Learned SPI and used an adapted concept suitable to the organisation. Made the whole scope of the project visible. Maintained the organisation's view of process improvement.	Slowness of the initiative caused frustration and tiredness. Inertia of the improvement process.
	Norm	Seek guidance in ideal processes.	No	To focus on a specific area both for diagnosing the current problems and improving the software processes.	SPID used a modified CMM-based assessment to identify the current maturity level of the organisation without the aim to reach any maturity level in the CMM. SPID based the improvement strategy on the assessment findings and earlier experiences of software problems.	Gained management and practitioners' commitment to the project by "not just aiming to reach an abstract level in a model". Succeed in understanding the current problems and improved some software processes. Spend a great deal of time on modifying the CMM-based model.	Incoherent improvement. No long-term vision.
	Commitment	Ensure dedication and legitimacy.	Yes	To gain management's and practitioners' support and commitment.	SPID kept management and practitioners informed about the project's status and maintained their commitment by having continuously communication with both groups. SPID had the management's support and the practitioners' commitment to co-operate in the creation of new software processes during the whole project.	Ensured resources for the project. Developed a good platform for implementation. Spending too much time on one specific problem.	Time consuming.

Perspective in SPI	*Process*	Integrate people, management, and technology.	Yes	To agree upon processes including roles, tasks, and deliverables.	SPID defined processes by first focusing on the products of the process, i.e. the documents; the focus then changed to identify the activities and development tasks necessary for creating the documents. SPID as the next step focused on roles responsible for each task and then defining the process in focus.	Achieved identification of documents. Achieved definition of processes: Software documentation Software validation Software change and version control	Time consuming.
	Competence	Empower people through competence building.	No	To create and transfer SPI and software engineering knowledge to all practitioners.	In SPID, practitioners' experiences and ideas were the main input to the improvement activities. The practitioners shared their experiences in improvement meetings. The creation and transfer of SPI and software engineering knowledge has been limited to the SPID's organisation. The other practitioners have not yet become involved in the competence building process.	Empowered SPID's members through competence building. Facilitate the improvement activities. Facilitate the acceptance decision (management acceptation of the new processes).	Necessity of much effort to transfer the same knowledge to all other practitioners in the whole organisation. Time consuming.
	Context	Establish suitable efforts.	Yes	To create a supporting SPI group helping software projects to adapt the newly created processes and continuously improve the processes on the basis of practical use.	At the time of writing in March 2001 SPID had created a support organisation including three SPI specialists to assist software projects in adapting the software processes in their projects; the members of the supporting group act as coaches, not controllers.	Facilitated the adaptation of the new created processes in practice by having a support group.	Time and resource demanding.

Table 1 SPID activities based on the MAP

4.1 The management of SPID

The SPI activities are organised as dedicated efforts.

SPID was organised as a project with its own specific budget and resources. An initial improvement infrastructure, i.e. project organisation including roles and responsibilities, was established early in the project in April 1999. Organising SPID as a project gave the organisation the possibility to allocate resources to it as with any other project at DevIS. This helped to gain the management's commitment. It further helped to structure the project's organisation and the responsibilities. Organising SPI initiatives as regular software projects has been supported earlier by Johansen and Mathiassen (1998), Zahran (1998), and Arent and Nørbjerg (2000).

Organising SPID as a project brought problems as well. The people who worked in the reference group were very busy with other projects at the same time, which sometimes meant that someone would not deliver what s/he was responsible for at a meeting. This was a minor problem, however, because the practitioners' commitment to the project was very high and would deliver as soon as possible. All the members of the reference group were experienced project managers and software developers challenging software process problems every day. There was a strong desire to solve the problems once and for all. Another problem related to organising SPID was co-ordinating the resources and meetings with both the management and the improvement team. This was very time consuming. The action demanded a great deal of planning time.

All improvement efforts are carefully planned.

Early in the project an SPI plan was developed by the author to define the goals, deliverables, milestones, and schedules of the project. Planning the project so explicitly helped us (the SPID organisation) to understand what to do, when to do which activity and who should do the different activities in the project. This supported a focus on the project schedule and the deliverables. On the other hand, it sometimes caused stress, especially for the project manager, because he pressured himself to deliver results on time.

Another problem of this detailed planning was that the whole SPI group including the reference group and the SPI working group sometimes felt it was prisoner of the schedules and deliverables, yet it did not let its thoughts and ideas be stopped or disturbed by the details of the plans. The group tried continuously to "reflect-in-action" (Schön 1987) and changed some plans and some deliverables in a few cases. These changes were either necessitated by other ongoing improvement activities in the company that affected the project or a feeling in the group that the respective changes in a specific plan and its deliverables would yield better results in the end. The plan was to develop a framework for action rather than a procedure to follow in detail. Having a framework for action helped the group to be more motivated and engaged in the project.

Feedback on effects on software engineering practices is ensured.

Routines for gathering feedback about how to measure improvement in the newly created software processes were originally not defined explicitly as it was assumed that the reference group of experienced project managers and software engineers would give feedback on the improvements as the project went along, and so was the case. In the SPI plan it was simply stated that the results of the project, the new software processes, should be tested in two software projects before implementation in the entire organisation. However, during the implementation phase of the project (implementing the new processes in the organisation, June 2000-June 2001), a qualitative feedback measurement mechanism has been established based on the implementation plan of the project. This mechanism is gathering feedback on the use of the processes and the templates through three communication channels: 1) training sessions, by asking practitioners what they think about the new processes 2) customisation meetings, in which the SPI group helps practitioners to adapt the processes to their project, and 3) through email, by gathering practitioners' further improvement suggestions; specially according to practical use of templates in software projects (see Pourkomeylian 2001c).

The fact that no explicit routines for providing feedback on improvements were defined in detail did not seriously affect SPID's initiation, diagnosing and improvement activities. Between April 1999 and June 2000 the project focused on diagnosing the software project discipline, identifying current problems, suggesting improvement areas and improving a few processes. By not defining feedback the SPI group could spend time on other issues such as reading the SPI and software engineering literature and AstraZeneca's and the authorities' quality requirements. The problem was that the SPI group did not know whether the new software processes worked in practice; it was not possible for the group to measure and document quality and progress. However, there were three experienced project managers and two software engineers in the reference group who had experience in software validation and change control. They provided continuous feedback on the new software processes during the project's life span. In the short term this created an informal control mechanism that checked whether the improvement activities were focused on the right solutions. In the long term, however not defining feedback routines can cause problems such as uncertainty about initiatives and misdirection of the project as a whole with the risk of not getting the desired results (Aaen *et al.* 2001).

4.2 Approaches to SPID

SPI is evolutionary in nature.

In SPID it was decided to improve a few software processes in an evolutionary way by taking one step at a time, which helped to concentrate on a few software processes at a time. As the SPI group could see the whole scope of the project, it did not get lost in the complexity of improving many software processes. Focusing on a few software processes did pose a problem, however, as these software processes were all part of the whole picture (which includes other processes such as: software test process and software configuration management process), which was beyond the scope of the project. The SPI group spent much time discussing the differences and the relations between life cycles, software development models, the software validation process and software change control. As the group continued its discussions it was noted that there were more areas that needed improvements. The members of the SPI group still regularly needed to remind each other of the scope of the project to keep focusing on the main goal of the project and not try to solve all the problems in one shot. The performance increases were limited and not visible during the project. As it could not be measured, the extent to which these processes were useful in practitioners' work was unknown. Today (March 2001) we continuously get feedback from the projects using the processes and the templates. This and the lengthy discussions on reaching agreement on a common process sometimes created a feeling in the group of burnout and not to being able to maintain its commitment to follow an evolutionary approach. As we all were aware of the nature of an evolutionary approach and were interested in solving our software process problems we kept maintaining our commitments to the project. Aaen *et al.* (2001) have also reported this issue to be a pitfall for SPI efforts.

SPI is based on idealised, normative models of software engineering.

The SPID goals were neither to reach any maturity level of the CMM nor to improve many software processes. To understand the current level of software process problems a modified CMM-based assessment was adopted and carried out in the organisation. As the goal was not to reach a maturity level, it was decided not to follow the normative CMM recommendations for improvement activities. Rather than trying to fulfil the requirements of the CMM, the SPI group derived inspiration from the concept of SPI, namely improving software processes in a systematic way.

Both management and practitioners reacted positively to the fact that the project did not aim to reach a certain maturity level of the CMM. Reaching a level in a normative model was too abstract for these groups at that time. However, solving the organisation's problems by seeking inspiration in a well-established model was appreciated. Still a great deal of time was spent on adopting the assessment to fit the organisation's terminology.
Focusing on solving the organisation's problems rather than simply following a model to reach a level led to greater motivation and enthusiasm among practitioners and management whether the improvement activities were focused on the right solutions. In the long term, however not defining feedback routines can cause problems such as uncertainty about initiatives and misdirection of the project as a whole with the risk of not getting the desired results (Aaen *et al.* 2001).

SPI is based on a careful creation and development of commitment between the actors involved.

SPID's improvement strategy, to proceed in an evolutionary manner, was based on the assessment's findings, company's quality goals, and practitioners' and management's commitment to the project. A key factor in succeeding to improve the new processes was that management and practitioners both were committed to the project and concerned about its results. This commitment was gained and maintained through continuously planned meetings to inform and discuss the concepts of SPI and software engineering and the progress made in SPID during the project. With the management's commitment the project had a sponsor that dedicated the necessary resources to the project.

Even though commitment was vital for SPID, it was carried too far in some situations. The SPI group sometimes became so dedicated to solving problems that it lost sight of the original focus of the project and started to discuss other related issues such as life cycle and software development models. This led to a loss of time and created stress when deadlines approached. However, the SPI group members continued to remind each other of the goal of the project, that is to focus on improving the software documentation process, the software validation processes, and the software change control and document version control processes during their meetings and tried to avoid loosing that focus. Aaen *et al.* (2001) have also reported this issue to be a pitfall for SPI efforts.

4.3 Perspectives in SPID

SPI is focused on software processes.

At the time the SPID started in April 1999 DevIS lacked a detailed description of all software processes. This meant that different interpretations of any given simple software process activity existed in all the different software projects. For instance, practitioners knew that a software product should be validated before it was put into operation, but the interpretation of the software validation process varied in different projects.

From the earlier problem analysis activity performed in early 1999 it was known that a description of the software documentation process was lacking in the organisation. General knowledge about which documents should be created as products of software projects supported the identification of the tasks needed to create the documents. This knowledge existed in the organisation and in the SPI group in different interpretations and needed to explicitly be documented as an agreed common process. After the definition of the software documentation process the focus changed and was concentrated on two already existing processes that should be improved, namely software validation and software change control.

A benefit of focusing on the documentation process was that we agreed on one of the most important processes related to quality and compliance issues from both the company's and authority's point of view. Thus the creation of a software documentation process helped the organisation to view documents as one specific, explicit and important type of software product. The result of the process is summarised in a general documentation matrix that includes the names of all necessary documents and the names of the roles responsible for producing, reviewing, and approving each document. This matrix is now used by practitioners in software projects as a general documentation model for identifying all necessary documents to be produced in the project. The number of different documents needed in a project depends on several factors such as authority requirements, complexity and size of the software project, and the project's organisation. The documentation matrix was also used as a major input for the definition of the software validation and change control processes.

A problem of focusing on certain software processes was, as explained above, that the SPI group did not have the whole picture (a process map) including other software processes. It could not see the relations between the processes in the focus on improvements and other processes such as software configuration management or other models such as software development models. Many hours of discussion were required to separate different issues from the project's main target and to focus on the defined goals.

One problem that Aaen *et al*. (2001) mention in focusing on processes is the danger of losing the customer perspective. In SPID, we addressed the issue that the software developers in the reference group represented the user community and they gave their input to the process improvement activities from their own point of view.

The practitioners' competencies are seen as the key resources.

One input to the improvement work were the SPI group's competencies, ideas and experiences as well as the software engineering literature and the company's standard procedures. Still, the major practical input were the practitioners' - the reference group's software project managers' and software developers' - experience and ideas based upon their practice. In several meetings the practitioners working in the SPID shared their ideas about and experience with software documentation, validation, and change control. According to the members of the SPI group, their competence in both SPI and the area of software engineering developed during the course of the project. Although not all the practitioners could built up practical competence at that time, competence building was still a key part of the implementation activities as all practitioners were trained in new software processes and learned the new software processes by applying them and the new templates in their practical works.

One problem of focusing on practitioners' competencies to improve software processes was that the entire project was dependent on their input. If a majority of practitioners could not attend a meeting it had to be cancelled and the project stood still until it was possible to schedule another meeting. This problem has also been identified by Johansen and Mathiassen (1998).

Another problem in using practitioners' ideas and experiences as input to improvements was that the members of the reference group had different experience from different software projects. They had different interpretations of any specific software process or task. Much time was therefore needed to discuss different views and experiences related to each software task or process. There was however an advantage that we (the SPI group) knew that all the different interpretations of a given task or process that were discussed had already been used in practice and shown some degree of usability.

The meetings in which the SPID and the reference group discussed different interpretations of a specific task also supported experience sharing and competence building. However, during improvement activities, only the members of the project were able to make gains through competence building. Other practitioners were not actively involved in improvement activities and thus could not participate in sharing experience. There was a risk that the project members could become sort of elite group that had knowledge about the new processes. This might have had a negative impact on other practitioners and create resistance against using the new processes. Some effort has been put into giving information about the results of the project on a continuous basis throughout the project for keeping other practitioners informed.

SPI aims at changing the wider context of the software operation to create sustainable support for involved actors.

To support the implementation of the new software processes throughout the organisation, the SPID project created an infrastructure in which the processes could be implemented and continuously improved, although this was not part of the early deliverables defined in the project plan. The closer the SPI group came to the end of the originally scheduled improvement activities the more it realised the need for a support group to take over the new software processes and implement them in the organisation. This need was not clear to the SPID group at the start of the initiative. Now a supporting group including three SPI specialists helps with the adoption of the processes in every software project. This is a time and resource consuming process, but it has resulted in successful adoption of the new processes in 12 software projects, which is a small amount of all projects running in the company. Aaen *et al.* (2001) argue that there is a risk that a group like this could be experienced by practitioners as a controlling function or bureaucracy. To reduce this risk the members of the supporting group act as coaches and discussion partners to assist the software projects in adopting the processes rather than controlling the implementation of the processes in projects.

5 LESSONS LEARNED

This study analysed an SPI project based on the characteristic features of SPI as identified by Aaen *et al.* (2001). Having applied the framework to analyse an initiative in a beginner organisation, the author is convinced that a novice organisation can succeed in its first SPI initiation as DevIS did, if it gains and maintains management and practitioners' commitment, starts with a small focus on a few carefully selected software processes that need improvement, and uses the existing knowledge about the software processes in the organisation. This study has led to a number of lessons relevant to future SPI projects in novice organisations and the wider SPI practice and confirms the research which builds the basis for the MAP framework.

Lesson one: *A first SPI initiative should always be organised as projects with specific goals, deliverables, and resources.*
Organising the SPI initiative as a project and having a flexible plan as a framework for conducting the activities was essential for managing improvement activities in our first SPI initiative. By carrying out the SPI effort as a project the initiative received the same status as all other software development projects in the organisation. This – automatically - led to resource dedication and management commitment, which were crucial factors for the SPI project as they are in all other projects. Organising the SPI effort as a project required some degree of administration and time. A detailed plan including goals, schedules, roles, responsibilities, and deliverables was created and a project organisation including project manager, steering committee, reference group, and a working SPI group was established. This helped the project members to be able to see the start and the end of the first SPI initiative and to have a common understanding of deadlines and deliverables in the project.

Lesson two: *A novice organisation should focus on the SPI concept rather than on model-based recommendations like those of the CMM.*
A first SPI initiative as all succeeding ones should start by diagnosing the problems and thus only implicitly the current maturity level of the organisation. A model like the CMM can offer much help in doing this. However, for improving software processes, it seems more promising to rely on the organisation's actual goals and to focus on the related SPI activities and to improve a few software processes based on these organisational goals rather than simply follow a model like the CMM. This happened in the SPID project and helped gain management and practitioners' commitment to the project so that there was an appreciation of SPI as a systematic concept and a background for improving the software processes. In the long run, however this kind of strategy might lead to a lack of an overall long term vision for continuous software process improvement initiatives. Here, more direct guidelines from models like the CMM might be helpful.

Lesson three: *It is highly recommended that a MAP analysis be made early in the initiation phase of the first SPI initiative to understand the most characteristic features of the SPI project.*
AstraZeneca did not have access to the MAP framework at the start of the SPI initiative. The MAP analysis was performed little over half way through the project to investigate whether any important features were missing in SPID, to reflect about the course of the initiative and to eventually take corrective action. As a result of the MAP analysis, we are planning a feedback measurement

activity based on the results of practical use of the processes in these 12 software projects. We have also initiated discussions with management for long-term improvements.

An analysis early in a SPI effort might provide better opportunities for a novice organisation to understand the most important features of its first SPI initiative and as a consequence might lead to better planning, organising and risk management of the first SPI activities.

ACKNOWLEDGEMENT

This research was sponsored by AstraZeneca R&D Mölndal in Sweden in collaboration with the Viktoria Institute Gothenburg in Sweden and the Department of Computer Science at the University of Aalborg in Denmark.

REFERENCES

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement. *Scandinavian Journal of Information Systems,* Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 123-146.

Andersson, I. and Nilsson, K. (2001). Diagnosing Diffusion Practices within a Software Organisation. *IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Arent, J. and Iversen, J. (1996). *Development of a Method for Maturity Assessments in Software Organizations based on the Capability Maturity Model.* Aalborg University: Department of Computer Science.

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science.*

Curtis, B. (1994). A Mature View of the CMM. *American Programmer,* Vol. 7, No.9, pp. 19-27.

Goldenson, D. R. and Herbsleb, J. D. (1995). *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-009.

Hayes, W. and Zubrow, D. (1995). *Moving On Up: Data and Experience Doing CMM-Based Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-008.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. C. (1997). Software Quality and the Capability Maturity Model. *Communications of the ACM,* Vol. 40, No. 6, pp. 30-40.

Humphrey, W. (1989). *Managing the Software Processes.* Addison-Wesley.

Johansen, J. and Mathiassen, L. (1998). Lessons learned in a National SPI Effort. EuroSPI 98 Gothenburg, Sweden, November 16-18, pp. 5.1-17.

Kuvaja, P. and Bicego, A. (1994). BOOTSTRAP - a European assessment methodology. *Software Quality Journal*, Vol. 3, pp. 117-127.

Mashiko, Y. and Basiili, V. R. (1997). Using the GQM Paradigm to Investigate Influential Factors for Software Process Improvement. *Journal of Systems and Software*, Vol. 36, pp. 17-32.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

McFeeley, B. (1996). *IDEAL. A User's Guide for Software Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Handbook CMU/SEI-96-HB-001.

Paulk, M., Curtis, B., Chrissis, M. and Weber, C. (1993). *Capability Maturity Model for Software Version 1.1.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-96-TR-24.

Pourkomeylian, P. (2001a). Knowledge Creation in Improving a Software Organization. *Proceedings from IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Pourkomeylian, P. (2001c). An Approach to Institutionalisation of Software Processes. *Proceedings from Tenth International Conference on Information Systems Development*, London, England.

Schön, D. A. (1987). *Educating the Reflective Practitioner*. Jossey-Bass Publishers.

Tryde, S., Nielsen A. D. and Pries-Heje, J. (2001). Implementing SPI: An Organizational Approach. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice*, Addison-Wesley.

Zahran, S. (1998). *Software Process Improvement: Practical Guidelines for Business Success.* Addison-Wesley.

AN APPROACH TO INSTITUTIONALISATION OF SOFTWARE PROCESSES

Pouya Pourkomeylian

Abstract: Software Process Improvement (SPI) has been shown to be a useful approach for improving an organisation's software capabilities. One great challenge for organisations wrestling with SPI is how to institutionalise the newly created software processes in the whole organisation. Institutionalisation of software processes is a matter of cultivating organisation-wide change, which has proven to be no small challenge for many organisations. Different factors such as the complexity of organisational change and culture, the human factor (resistance to change) and lack of a supporting organisation for change have caused problems in making change happen in organisations. These problems may cause delays, poor quality, cost overruns in software projects and even resistance to using new created software processes. This paper presents the results of a collaborative research study in which an institutionalisation model was created and implemented and analysed on the basis of a framework of organisational change management.

Keywords: Software Process Institutionalisation, Organisational Change Management.

1 INTRODUCTION

The Software industry has had a focus on Software Process Improvement (SPI) for a number of years. SPI was developed by the Software Engineering Institute (SEI), inspired by the work of Watts Humphrey (Humphrey 1989). The concept of SPI has caught on in many organisations that have started SPI initiatives for improving the capabilities of their software processes. An SPI initiative starts with an assessment to establish the maturity level of the organisation. The most popular assessment model is the Capability Maturity Model (CMM), which was also developed by the SEI of Carnegie Mellon University (see Paulk *et al.* 1993).

The next step after making an assessment in an SPI initiative is to organise and plan for software process improvement activities based on the organisation's requirements and practitioners' ideas on improving the capabilities of the organisation. After conducting software process improvement activities, the main challenge for an organisation is to implement the newly created software processes into the entire organisation to become part of practitioners' daily work.

Many organisations do not succeed in their improvement activities, while others have difficulties with implementation and institutionalisation of new processes in the organisation (Tryde *et al.* 2001). In general one can consider that different factors such as scaling the SPI initiative, setting realistic goals, the complexity of organisational changes, and the organisational culture have caused difficulties in succeeding in SPI initiatives (Goldenson and Herbsleb 1995, Herbsleb *et al.* 1997, Mashiko and Basili 1997, Johansen and Mathiassen 1998).

Without software processes implemented across the organisation, every practitioner will follow his or her own way of carrying out the task (Zahran 1998). Adherence to a common process is likely to be ad hoc and sometimes chaotic. This may cause delays, poor quality and cost overruns in software projects. On the other hand, when a practitioner performs tasks naturally and painlessly by following an adapted process, one might deduce that such a person is a professional. He or she has been properly trained to follow a well-defined and adapted process that is performed repeatedly. According to Zahran (1998), such a person has “internalised” the process and is capable of performing the process activities professionally and “painlessly”. While process internalisation is at the individual level, process institutionalisation is at the team and organisation level and concerns ensuring that software process improvement is continuous and is embedded in the organisation’s culture. Software process institutionalisation is a matter of building an infrastructure and culture that support methods, practices, and procedures in the whole organisation (Paulk *et al.* 1995). This will involve creating new roles and responsibilities, continuing management sponsorship, introducing new organisational policies and procedures, and establishing the measurement and enforcement mechanisms.

Implementing newly created software processes in an organisation is a matter of changing the current way of working and bringing about a new way. The management of change is of critical importance for the success of implementing new software processes in an organisation. One of the main reasons for its criticality is that it involves a wide spectrum of domains that may need to be changed, such as: cultural changes, behavioural changes, organisational changes, technological changes, and environmental changes. According to Burnes (1992) it is shown that theories that underpin models of change management can be distinguished by their respective concentration on individual, group and organisation wide issues. These levels have been in focus in the SPI literature through “internalisation” and “institutionalisation” of software process implementation Zahran (1998). According to Burnes (1992) and Weinberg (1997) change comes in many shapes and sizes, though most forms can be categorised as either radical or incremental. Radical change often relates to large-scale, organisation wide change programmes involving the rapid and wholesale overturning of old ways and old ideas and their replacement with new and unique ones. Radical change is however characterised by its speed, scale and break with the past. On the other hand, an incremental change process can only bring an ad hoc, local improvement in performance (Burnes 1992). It is clear that changing, even in a small way, can be complex and difficult. The literature abounds with examples of changes that have gone wrong, some disastrously so (Burnes and Weekes 1989, Kelly 1982).

One main reason for not succeeding in implementing change in organisations has been the factor of resistance to change. The instinctive human reaction to change is usually a rejection of the new change and preference for the status quo. This rejection may cause problems in implementing change in organisations (Zahran 1998, Weinberg 1997, Beer 1987, Jacobsen and Thorsvik 1989). Our experience tells us that, because of the rejection to change and for other reasons, many process improvement activities do not lead to a full internalisation or institutionalisation of the new processes in the organisation. Practitioners’ resistance to change can have different reasons. Jacobsen and Thorsvik (1989) mentioned some factors that can cause resistance to change, such as: *Expectations*: a change in the working processes fails to satisfy practitioners’ expectation; *New knowledge:* implementing the newly created processes in the organisation might require individuals to have new knowledge; *Risks and insecurity:* using the new processes can bring risks and lead to the creation of an insecure environment in the organisation; *Power:* changing from one way of working to another might change the stable power and influence balance in the organisation and therefore cause resistance to change. Other reasons for resistance to change might be: *Wrong process level:* the level of the software processes is not suitable to the situation (they are too complex or too simple); *Not being involved in the improvement process*: the new processes have just come from the top and non of the practitioners has been involved in the improvement activities;

Bad timing: the implementation time is not suitable, practitioners have no time to become involved in a change process; *Lack of supporting infrastructure*: lack of a support for the practical implementation of the processes in the organisation.

On the basis of our experience we know that these factors might cause two main type of behaviour among individuals in an organisation: 1) Total resistance to use of and complete failure to use the newly created processes, 2) Partial use of the processes in some cases with individual interpretations (which often are not complete or correct) of them and use of them as long as this does not cause problems (extra work) in the software projects. Another completely different behaviour might be that practitioners create routines for the software processes or standards and insist on following them in detail, causing bureaucratic behaviour and leading to inertia and surprises in new, unknown situations outside the boundaries of the processes. On the basis of our experience, we know that such behaviour can lead to problems in software projects, such as: delays, poor quality, cost overruns, and even distrust in management and future SPI initiatives. We agree with Weinberg (1997) that, because human systems do not change unless the individuals change, and individual's commitment to change comes through understanding and learning the new situation. Therefore, to understand the nature of resistance to change and the practice of institutionalisation and internalisation, we have focused in this study on the emotional reactions of practitioners and the need of an infrastructure as a facilitator of change.

Using a change framework based on four change models described by Weinberg (1997), this study analyses one institutionalisation approach in an SPI project in a software organisation. The attempt was find answers to one main question: what are the most crucial factors in a software process institutionalisation model from an understanding and learning perspective?

The section below presents the research approach. Section 3 presents the case and the research project. Section 4 describes the theoretical concept including software process institutionalisation, change management, and reflection-in-action. Section 5 presents the institutionalisation model. Section 6 discuses the theoretical interpretation made on the basis of our findings, and section 7 concludes the paper by presenting the lessons learned and pointing out areas for further research.

2 RESEARCH METHOD

This study uses a collaborative practice approach (see Mathiassen 2000) to analyse an institutionalisation model in an SPI project done in a software organisation. In this study we have combined action research with case studies and field experiments. The action research approach involved the author in gaining first-hand experience from the SPI project, the software organisation, and the institutionalisation practice and allowed him to develop knowledge and understanding as part of practice. The action research approach is chosen for this study because of its strong support in integrating research and practice, involving practitioners in the institutionalisation practice and introducing change at the same time that the research was going on in the software organisation. Case studies are particularly useful for understanding a special subject in great depth, where one can identify cases rich in information, in the sense that a great deal can be learned from a few examples of the phenomenon in question (Patton 1990). This approach was chosen for this study to help us focus on some cases (software projects using newly created software processes) and to provide us with techniques to structure the research process and its findings. Field experiments in this research were set up as controlled research efforts in which different elements of our institutionalisation model were implemented in the entire organisation. The institutionalisation efforts included trainees for all practitioners in the organisation, implementing the processes in ten software projects qualitative measurements of the effects of the processes and work with continuous improvement.

The author has been the driving force in the primary SPI initiative that led to the creation of the software processes and has actively participated in activities to initiate, organise, plan, and conduct the SPI initiative over a period of two years (see Pourkomeylian 2001). He was also the driving force in the creation and implementation of the institutionalisation model in the software organisation. In this study the author reflects on the institutionalisation process and tries to make conclusions about lessons useful for understanding the field of software process institutionalisation from the perspective of implementing change.

3 THE CASE

This study was conducted as part of an SPI research project started in 1999 at AstraZeneca R&D Mölndal in an IS organisation called Development IS (DevIS) aiming to improve the capability of the software organisation. AstraZeneca is one of the world's leading pharmaceutical companies with a formidable range of products designed to fight disease in important areas of medical need. DevIS supports clinical and pharmaceutical projects, Regulatory Affairs, and Product Strategy and Licenses at AstraZeneca R&D Mölndal. DevIS is also responsible for influencing the development of the global clinical research processes and IS/IT tools at AstraZeneca.

DevIS employs 100 people including contractors, most of whom have backgrounds in IS/IT. Employees of DevIS are basically engaged in software development, software maintenance, and operation activities. The software development activities occur in two forms: 1) development of totally new software products (software development) and 2) developing or changing existing software products (software maintenance). A typical software development project at DevIS is scheduled to take between six months and one year and includes analysis, design, construction, testing, implementation, and validation. Software maintenance activities can consist of changes in the code or developing a completely new application for existing software products. Software products in DevIS include the software and all related documentation (e.g. user requirement specification, test plan, validation plan, validation report, user manuals etc.).

The SPID Project

The results of a problem analysis done in early 1999 in one of DevIS' largest software development groups showed a need for improving software project discipline and providing guidelines for understanding the standard operation procedures addressing different authorities' quality requirements. The director of DevIS initiated an improvement project called Software Process Improvement at DevIS (SPID, headed by the author as project manager) whose purpose was to understand the existing problems and improve the organisation's software processes. The first phase of the SPID project was initiated, organised, planned, and performed during the period from April 1999 to May 2000 and intended to improve DevIS's software processes.

A maturity assessment using a modified CMM based (Capability Maturity Model) assessment method (QBA, see Arent and Iversen 1996) was done and showed that DevIS was by then a level one organisation and addressed improvement possibilities in all analysed KPAs (Key Process Areas, level 2). An improvement report based on the assessment's findings and other findings from earlier improvement initiatives at DevIS addressed six improvement activities. The steering committee of SPID gave priority to the following improvement activities: 1) To establish a minimum documentation level for documenting the results of software projects and create the software documentation process. 2) To improve processes for software validation, software change management, and document version control. 3) To create a template library including templates for documentation of software development activities, such as: user requirement specification, design specification, test plan, and validation plan.

An improvement plan was created and an SPI-group (including the author, two SPI consultants and five software engineers) was created and started planning and performing improvement activities over a period of four months, which resulted in creation of new software process guidelines. Based on the results of a performed Participatory Implementation Workshop (PIW) see (Andersson and Nilsson 2001) an implementation plan was created which was the basis of the creation of an institutionalisation model. The aim of the workshop was to based on the software organisation's requirements together with some practitioners from the software organisation suggest an adapted institutionalisation model for implementing the newly created software processes in the organisation. After presentation of the newly created software processes and the institutionalisation model for the management the steering committee of the project accepted the new software processes and decided to implement the newly created processes within the whole organisation. The implementation activities are scheduled to be arrayed out between June 2000 and June 2001.

4 THEORETICAL CONCEPT

4.1 Change Approach

The choice of change approach depends on the organisation's nature, resources, and problems (Zahran 1998). An organisation may thus adapt one or more change approaches to drive change. Weinberg (1997) presents different change approaches with different effects in changing organisations daily work.

The Diffusion Model
This model focuses on change as something that simply happens. According to this model, in many instances, a change seems to come about throughout an organisation without any specific management action (no planning and no control, change just happens when it happens). The strength of this model is its attention to change as a process. The weakness of the model is the abdication of control over that process to a passive, mysterious "force of nature".

The Hole-in-the-Floor Model
The Hole-in-the-Floor, or Engineering, Model attempts to correct the weakness of the Diffusion Model by adding control of the change process. According to this model experts develop the "perfect" solutions and the change plan consists of "drilling a hole in the floor". The new solutions will then be "dropped" through the hole with the intent that practitioners will use it happily ever after. The difference between this model and the Diffusion Model is that, according to this model, change happens if and only if all preparations are correct. This model is often proffered by experts who believe that organisations behave logically i.e. that everyone will undoubtedly recognise the benefits of their proposal to change, immediately accept the proposed change and be willing to change the way they work. Its strength of this model is the emphasis on planning. Its weakness is that the planning leaves out many essential factors, most notably the human factor.

The Newtonian Model
One way to introduce the human factor to the hole-in-the-floor model is to use the Motivational Model, which has also been called the Newtonian Model (named after Isaac Newton's laws of motion). This model predicts that change happens faster when you push harder. The larger the system you want to change, the harder you must push. Likewise the faster the change you want, the harder you must push. Force and acceleration are two factors that have directions. The model thus implies that, to change in a certain direction, you must push in that direction. According to this model, what is missing in the hole-in-the floor model is the push. In this respect the model does recognise that people have a choice in what they do and that their choice can be influenced (by pushing them) as part of the change process. Typical pushes include offering bonuses, threatening loss of jobs, or rewarding with challenging assignments. But one should remember that push works both ways. Many changes are set up to fail because the force pushing for them is overbalanced by other forces that push against them. The strength of this model is the explicit introduction of the human element in the form of motivation. The weakness is the totally inadequate model of humanity that's used: that people can be pushed around like billiard balls.

The Learning Curve Model
It has been observed that people are not usually able to respond with instant efficiency when change is first introduced. Moreover, once they do respond, it takes time to learn to respond as well as the planners would hope and thus to realise the intended benefits of the change. This model predicts that change occurs along a curve characteristic of the people making the change. The curve is obtained by averaging performance over many individuals and thus may smooth out significant individual variations. The model says that all changes follow some sort of learning curve. Moreover, the actual values of the curve are affected by a number of psychological factors, such as relevant skill, motivation, and aptitude. This suggests the possibility of influencing the course of the change by personnel selection and training, which certainly represents a set-up in realism as in the Newtonian Model. This model is quite useful for predicting the time scale of large-scale change but it does not go far enough as a practical tool for managing change person-by-person in a real organisation. The strength of the model is its incorporation of the adaptive human element in change. The weakness is the averaging out of details of individual human beings.

In summary, what these models together predict is that change should happen as a controlled process pushed by management through detail planning focusing on teaching practitioners new processes on a group level.

4.2 Reflection-in-Action as a Facilitator to Change

The newly created software processes in an SPI initiative have the character of "facts", "procedures", "rules", and "theories", which all are static. The documented software processes can be seen as explicit knowledge (see Schön 1987, Nonaka and Takeuchi 1995, Nonaka and Kanno 1998, Arent Nørbjerg 2000, Pourkomeylian 2001).

If we assume that all the practitioners in an organisation are willing to use this explicit knowledge in their daily work and that the management is also committed to supporting the institutionalisation process for implementing the newly created processes in the entire organisation, the main challenge will be to make practitioners understand and learn how to use the processes in practice (Nonaka and Takeuchi 1995, Schön 1987). To do this, the practitioners need what Schön calls *knowing-in-action*, which is dynamic and refers to the sort of know-how we reveal in our intelligent action publically observable or physical performance. In both cases, the knowing is *in* the action. Knowing-in-action is tacit (see Schön 1987, Nonaka and Takeuchi 1995) but it works as long as the situation falls within the boundaries of what we have learned to treat as normal. We all desire to structure this knowledge and reshape it and make routines for using it to do our daily tasks faster and easier. When we have learned how to do something, we can execute a smooth sequence of activity, recognition, decision, and adjustment without thinking about it. But a familiar routine can produce an unexpected result, such as that an error simply resists correction or the usual actions do not produce the usual outcomes.

We find something odd about these situations because, for some reason, we have to look at them in a new way. Such experiences, pleasant and unpleasant, contain an element of *surprise*. Something fails to meet our expectations. In an attempt to preserve the constancy of our usual patterns of knowing-in-action, we may respond to surprise by brushing it aside, selectively ignoring the signals that produce it. Or we may respond to it by reflection. According to Schön (1987) we may do so in one of two ways. We may reflect *on* our action, thinking back on what we have done in order to discover how our knowing-in-action may have contributed to an unexpected result or we may stop in the midst of our action and think (see Arendt 1971). In non-of these ways, our reflection has no direct connection to present action; we do not interrupt the process of our action. In an action-present, a period of time that is variable with the context and during which we can still make a difference in the situation at hand, our thinking serves to reshape what we are doing while we are doing it. In cases like this, according to Schön (1987), we *reflect-in-action* to change a present situation.

5 THE INSTITUTIONALISATION APPROACH

The institutionalisation approach used in this study was created through a Participatory Implementation Workshop (PIW) (see Andersson and Nilsson 2001). Two external consultants were invited to hold a PIW workshop at the software organisation to help us identify the most important factors needed for creating an implementation strategy for the newly created software processes. The PIW workshop was conducted on May 18, 2000, at AstraZeneca. The author, one SPI consultant, one project manager (all involved in the software process improvement efforts that led to the creation of new software processes), and the two external PIW consultants participated the workshop. The group together approached several important areas and discussed: the scope of the implementation, what is to be implemented (the product), the target groups, the success criteria, roles and responsibilities, the organisation of implementation activities, the implementation activities, and the implementation strategy. The workshop ended with a risk analysis identifying the most important risks and the measures needed for them.

The group agreed on an implementation strategy that meant a “direct introduction” of the newly created software processes in the entire organisation with tests of processes in some pilot projects. The reason for choosing the “direct introduction” approach was that no software process guidelines were in use at the time except for standard operation procedures. The new software processes were created on the basis of the ideas and experiences of experienced project managers and software developers (see Pourkomeylian 2001). The processes were already needed in some software projects and used by two others in the software organisation. The consequences of choosing such a “direct introduction” strategy were that we had to: Plan very carefully to make everything work the same day that improved software processes were introduced. Ensure that all necessary roles and functions were in place. Communicate new changes to the entire organisation. Ensure that the supporting organisation was in place. Plan and carry out the training program.

On the basis of the results of the PIW workshop the author developed a report for the implementation of the new software processes in the organisation. The report included the institutionalisation model, detailed description of all roles and responsibilities, the main steps needed to make the institutionalisation process complete, time schedule, costs, and an organisation for creating an infrastructure for supporting the whole process. The report was presented to the management on May 29, 2000. The management agreed upon the content of the plan and decided to implement the new software processes in the whole organisation. This meant that the processes were to be institutionalised in the entire organisation within one year and that the second version of the software processes would be delivered by the end of June 2001.

Our institutionalisation model consists of three main elements: 1) a training program for all practitioners in the organisation, 2) process adaptation for all software projects, and 3) feedback measurement and continuous improvements of software processes. The heart of our institutionalisation model is the Software Process Improvement Unit (SPIU). This unit is specifically responsible for the development and maintenance of the organisation's software process guidelines and consists of the author and two SPI consultants responsible for arranging and offering training to all practitioners in the organisation. The unit also helps every software project to adapt the new software process guidelines into the project and correctly interpret the contents of the standard operation procedures and follow up the software projects. The SPIU is also responsible for measuring feedback and continuously improving the software processes and standard operation procedures. The continuous improvement activities according to our model will result in the creation of the second version of the software process guidelines.

5.1 The Training Program

Human systems do not change unless the individuals in it change, and an individual's commitment to change comes through understanding and learning the new situation (Weinberg 1997). Our first initiative to reduce resistance and motivating practitioners in using the processes was therefore to introduce the software process guidelines, the institutionalisation model, and the contents of the standard operation procedures to them. We arranged a training program to develop the skills and knowledge of practitioners so they could do their work effectively and efficiently. Our training program involved identifying the training needs of the organisation, software projects, authorities' requirements, and practitioners. We then developed and carried out training to address the needs identified. The training program was held during October 2000.

Our target in the training program was to open a window to start communicating with practitioners, line managers, system owners, and quality assurance (QA) people and to gain their trust while introducing the software process guidelines to them and teaching them the main general message of the new processes. We also wanted to present the SPIU to them as a coaching function that was supposed to help software projects in using the software process guidelines in practice. The training program consisted of three occasions for software process guideline trainees and three occasions for standard operation procedures, three hours for each occasion. Three one-hour training sessions were also offered to managers and system owners. A total of 107 persons received training by the SPIU. On the training occasions we sought feedback from the practitioners about software process guidelines and how the processes correspond with their way of working. These ideas were then put into an improvement database as potential improvement possibilities. We also measured practitioners' understanding of: the role and purpose of the SPIU, their own roles and responsibilities, the standard operation procedures, and the software engineering process before and after the training. It showed that the training sessions had helped to raise practitioners' awareness of the following aspects.

5.2 The Process Adaptation

Another initiative to reduce resistance and motivate practitioners to use the software process guidelines was to create an infrastructure for supporting the practical use of the software processes. By practical use of these processes we mean that every software project should be able to easily adapt the processes and use them in the daily work in his or her project. We knew by experience that if practitioners do not understand the processes, or if they have doubts as to whether they have made the correct interpretation of the processes, then the risk of not using or misusing the processes will be high. Another risk was that they could see the processes as the sole truth and simply follow them blindly in every situation. The problem was then the surprise of not being able to solve unknown situations outside the boundaries of the defined processes (see Schön 1987).

Our answer to these problems was the SPIU itself and the infrastructure we created to adapt the processes into every software project. To gain practitioners' trust and motivate them to use the help offered by the SPIU we started to act as coaches. This meant that we became a helping function, a supporting, advising function for the software projects rather than a quality control group. To do this we needed to be very familiar with the content of both the standard operation procedures and the software process guidelines, to understand the business and our software projects, and to be able to open and maintain a dialogue with the project managers and developers working in the software projects. To succeeding in our role as coaches and not controllers, we needed to see both the standard operation procedures and the software process guidelines as frameworks, not as procedures that should be followed blindly. We treated each project as a unique case with unique requirements and conditions. We analysed every project on the basis of its specific conditions, such as size, project organisation, schedules, authority requirements, and complexity, and created a specific process design for every project, which was approved by the project team and the SPIU within the boundaries of the standard operation procedures and the software process guidelines.

We did this through so called *customisation meetings*. The meetings consisted of two parts: an introductory meeting and follow-up meetings. The introductory meetings aimed to establish a documentation matrix based on the project's conditions in collaboration with the project manager and one developer in the software project. The documentation matrix showed which document should be created as a product of the software project. The matrix also showed what role should be responsible for the creation, review, and approval of the documents and addressed the activities necessary to create these documents. The meetings often took two hours and resulted in a documented agreement between the SPIU and the project manager about how to work in the project. At the introductory meetings we discussed questions such as how to validate the software, how to manage changes, and how to document the results in the specific project. The follow-up meetings aimed at measuring the extent to which the project had succeeded in following the tailored processes. Further, these meetings aimed at checking for problems in following the processes in practice and finding out whether these problems occurred in other projects, which might be an indication of a common problem and thus something that should be improved. The SPIU also helped the projects with reviewing the documents created during the projects' life span. In this way we created a frequently used communication channel with all projects which used our services in the organisation and tried to encourage practitioners to reflect in their actions during the projects' life span and tailor the processes to their software projects in a way that suited their conditions. We encouraged every project to make variations in the processes to adapt them in every project within the boundaries of the standard operation procedures and software process guidelines.

The customisation meetings helped us to see how these processes actually work in practice and motivated the practitioners to be more positive toward using the SPIU and the software process guidelines. The following are some comments made by the practitioners in the customisation meetings:

" A good way of working, to adapt the documentation process to our little project."
"These templates are most suitable for use in "real" software development projects, in our case we need to change or take away some headlines".
" We need more support for the Training Plan document. We do not know what should be written there".
" The templates work well as check lists".

The majority of variations made to adapt the software process guidelines in software projects were in creating different documentation matrixes for different projects. In small projects we often combined two or three documents into one document. Another type of variations was caused by differences in the project organisations. In small projects one person often had more than one role and responsibility. For both small and large projects, however, the same steps were taken in doing change control and validation.

We tried to implement the software processes in the software projects by encouraging the practitioners to reflect-in-their actions to continuously adapt the software processes and the standard procedures into the different situations in the software projects. The practitioners in the software projects thus learned the new processes by doing them in practice. By reflecting-in-action, they actually used their knowing-in-action for doing the "work", instead of just following the "routines". They reflected-in-action to change the practice in order to manage the unknown situation outside the defined routines they normally followed. By reflecting-in-action and using the guidelines and standards as a framework the practitioners created "variations-in-action" in practice different from case to case depending on situation in the software project and brought about continuous improvement in their daily work processes. But this was not an easy task neither for the SPIU nor for the practitioners. One big challenge for the SPIU was to continuously communicate the idea of reflection-in-action with the practitioners and encourage them in doing it in practice when interpreting the guidelines and standards as frameworks.

5.3 Feedback Measurement and Continuous Improvement

The institutionalisation processes were the first time the new software processes were presented to the practitioners and put into practice. In the training program and the process adaptation period we noticed that some activities in the processes and some headlines in some templates needed changes and/or improvements. For this reason we added a qualitative measurement mechanism to our institutionalisation model to capture all signals that addressed needs for improvements. With this mechanism we could make qualitative measurements at three main points: 1) during the training program, by asking practitioners about their ideas about the possibility of using the processes in practice, and 2) during process adaptation, by implementing the processes in software projects and following and measuring the results, 3) by practitioners sending email to the SPIU. We documented all possible suggestions for improvements voiced at the three measurement points and checked them during the practice implementation of the processes in different software projects. A typical improvement suggestion was as follows:

" Some headlines in the templates are not clear enough. You don't know whether you are writing the right text under the right headline".
"We need some examples to know for instance what a test plan should look like."
"How should we create user requirement specifications? What are the steps needed?"

6 THEORETICAL INTERPRETATION

This study analysed an institutionalisation process from an organisational change perspective using an institutionalisation model. From the perspective of this model we argue that change should not be forced or pushed in the organisation. It should be cultivated. Our approach differs from the more traditional approaches described in the theoretical part of this paper. We decided to start change in small areas and spread it step by step to all parts of the organisation. This approach is reminiscent of the diffusion model (see Weinberg 1997) but differs from it in that our model included factors such as planning, control, support, and measurement and continuous improvement. We did not simply drop changes into the organisation, as in the hole-in-the-floor model (see Weinberg 1997). We kept awareness of the human factor and took care of the important issue of resistance to change. We established different communication channels with the practitioners and helped them in practice by being there when they needed help. We did not take the role of an expert or a controller. We chose to be coaches for practitioners and to support them with their problems in using the new processes. Our ambition was to motivate the practitioners to use the processes by showing them the practical use of the system. We never tried to push or use management directives as forcing factors to drive the change unlike the Newtonian model (see Weinberg 1997) we encouraged the practitioners to reflect not only *on* their actions but *in* their actions for changing the ongoing practice in order to adapt the processes to their specific situation. In this way, it was possible for practitioners to avoid just following a routine and being surprised when facing unknown situations. We encouraged them instead to reflect-in-action (see Schön 1987) and act to create variations in order to adapt the newly created software processes to their specific software projects. Our model even completed the learning curve model (see Weinberg 1997) by supporting the individual learning of practitioners during the customisation meetings and individual support during the period of process adaptation into software projects. The learning process started on an individual level through training and continued to a group level (in software projects). In September 2000 only two software projects used our services. Today (March 2001) 11 software projects are clients of the SPIU using the software process guidelines. The SPIU has also gained organisational acceptance in the company. The processes are known among the practitioners and they are practising the idea of reflecting-in-action.

We could have speeded up the processes by using more "push" from the management. This was however a consciously strategy that we chose at that time. More "push" from the management may have negative effects. We let the practitioners learn and understand the benefits of using the processes and the supporting services of the SPIU in a more natural way rather than in a strictly formal way. The internalisation of the processes and even the implementation of software processes in software projects took its time but, on the other hand, the practitioners' attitude toward the processes is positive and the change has been cultivated and is maturing.

7 CONCLUSION

This study gives a number of lessons relevant for future software process institutionalisation projects and the SPI practice:

Lesson one: *Process institutionalisation should come through personal growth through learning and should start small to first reach process internalisation.* Our model focused on making the practitioners understand the new processes and learn how to use them in practice. We offered them training and made the processes and the templates accessible for them. We started the implementation process in the software organisation on an individual level through training and continued to a group level in a few software projects.

Lesson two: *Software process implementation should be carried out stepwise through detailed planning and follow-up.* Our institutionalisation model included a plan with a detailed description of roles, responsibilities, activities, schedules, resources, and deliverables. Without planning and following up the steps we could never have managed to carry out the different activities in a structured way.

Lesson three: *Management should encourage practitioners to reflect-in-action for creating variation-in-action, support the creation of an infrastructure to help in the adaptation of the software processes for every software project and address continuous improvement.* To support the practice implementation of software processes in all projects the management supported the creation of an infrastructure to help every project to adapt a specific process suitable to the project's requirements.

Lesson four: *New roles should be defined for coaching practitioners and teaching them new software processes on both the group and individual levels.* During the implementation process the change agents (the SPIU) had a coaching role. We offered training to teach the practitioners the new processes and to support learning on an individual level. We also supported learning on a group level by offering coaching services through the "customisation meetings" to trying teach practitioners how to go about in practice, for instance, validating their software product by letting them doing it and being there for them for advice and support.

ACKNOWLEDGEMENT

This research was sponsored by AstraZeneca R&D Mölndal in Sweden in collaboration with the Viktoria Institute Gothenburg in Sweden and the Institute of Computer Science Aalborg in Denmark.

REFERENCES

Arent, J. and Iversen, J. (1996). *Development of a Method for Maturity Assessments in Software Organizations based on the Capability Maturity Model.* Aalborg University: Department of Computer Science.

Andersson, I. and Nilsson, K. (2001). Diagnosing Diffusion Practices within a Software Organisation. *IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science*.

Arendt, H. (1971). *The Life of the Mind*. Vol. 1: Thinking. San Diego, Harcourt Brace Jovanovich.

Beer, M. (1987). *Revitalising Organisations: Change process and Emergent Model*. Harvard Business School.

Burnes, B. (1992). *Managing Change, A strategic approach to organisational development and renewal*. Pitman Publishing.

Burnes, B. and Weekes, B. (1989). *AMT: Strategy for Success?* NEDO, London.

Goldenson, D. R. and Herbsleb, J. D. (1995). *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-009.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. C. (1997). Software Quality and the Capability Maturity Model. *Communications of the ACM*, Vol. 40, No. 6, pp. 30-40.

Humphrey, W. (1989). *Managing the Software Processes*. Addison-Wesley.

Jacobsen, D. I. and Thorsvik, J. (1998). *Hur moderna organisationer fungerar, Introduktion i organisation and ledarskap*, Studentlitteratur, Lund, Sweden, (in Swedish).

Johansen, J. and Mathiassen, L. (1998). Lessons learned in a National SPI Effort. EuroSPI 98 Gothenburg, Sweden, November 16-18, pp. 5.1-17.

Kelly, J. E. (1982). *Scientific Management, Job Redesign, and Work Performance.* Academic Press, London.

Mashiko, Y. and Basiili, V. R. (1997). Using the GQM Paradigm to Investigate Influential Factors for Software Process Improvement. *Journal of Systems and Software*, Vol. 36, pp. 17-32.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

Nonaka, I. and Konno, N. (1998). The Concept of "Ba": Building a Foundation for Knowledge Creation. *California Management Review*, Vol. 40, No.3, pp. 40-54.

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

Patton, M. Q. (1990). *Qualitative Evaluation and Research Methods*. Sage Publications.

Paulk, M., Curtis, B., Chrissis, M. and Weber, C. (1993). *Capability Maturity Model for Software Version 1.1.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-96-TR-24.

Paulk, M., Weber C., Curtis B. and Chrissis M. B. (1995). *The Capability Maturity Model: Guidelines for Improving the Software Processes.* Addison Wesley.

Pourkomeylian, P. (2001). Knowledge Creation in Improving a Software Organization. *Proceedings from IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Schön, D. A. (1987). *Educating the Reflective Practitioner*. Jossey-Bass Publishers.

Tryde, S., Nielsen A. D. and Pries-Heje, J. (2001). Implementing SPI: An Organizational Approach. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice*, Addison-Wesley.

Weinberg, G. M. (1997). *Quality Software Management, Anticipating Change.* Vol. 4, Dorset House Publishing.

Zahran, S. (1998). *Software Process Improvement: Practical Guidelines for Business Success.* Addison-Wesley.

MANAGING KNOWLEDGE IN A SOFTWARE ORGANISATION

Lars Mathiassen & Pouya Pourkomeylian

Abstract. This paper explores the practical usage of insights on knowledge management (KM) to support innovation in a software organisation. The organisation has for some time engaged in software process improvement (SPI) initiatives to improve its operation. The paper applies two complementary approaches to KM, the codified and the personalised, to evaluate current KM practices and to improve its SPI practices. Based on the insights from the case we review key principles within SPI and evaluate the applied KM approaches. We conclude that it is advisable for SPI efforts to explicitly address KM issues. Each software organisation has to find its own balance between personalised and codified approaches, this balance needs to be dynamically adjusted as the organisation matures, and the adopted KM approach should differentiate between different types of SPI services.

Keywords: Software Process Improvement, Knowledge Management Strategy, Codification, Personalisation.

1 INTRODUCTION

Software projects are often late and users are often dissatisfied with the quality of software systems. This has for a long time been the public view of the software profession and most software professionals would agree. It is not surprising, therefore, that software organisations actively seek ways to improve their performance. The most widely used contemporary approach is called software process improvement (SPI). Many organisations pursue SPI initiatives and a number of success stories are reported (Humphrey *et al.* 1991, Brodman and Johnson 1995, Herbsleb *et al.* 1994, Goldenson and Herbsleb 1995, Haley 1996, Larsen and Kautz 1997, Mathiassen *et al.* 2001). But there are also many less successful cases (Goldenson and Herbsleb 1995, Debou 1997, Herbsleb *et al.* 1997, Mashiko and Basili 1997). This research effort seeks to increase our understanding of such innovation efforts in software organisations through application of approaches to organisational learning and change (Steltzer *et al.* 1998, Halloran 1999). More specifically we apply insights from knowledge management (KM) to inform SPI efforts in a large software organisation. The outcome is a revised improvement approach for the software organisation in question and general lessons on how to use KM insights to improve innovation in software organisations.

1.1 Software Process Improvement

SPI is a structured approach to improve a software organisation's capability to deliver quality services in a competitive way. SPI initiatives are evolutionary and inspired by experiences from quality management. An SPI initiative is cyclic in nature and includes different phases—1) Initiating, 2) Diagnosing, 3) Establishing, 4) Acting and 5) Learning—as expressed in the IDEAL model, see Figure 1 (McFeeley 1996). The initiating phase prepares the SPI initiative by developing plans, schedules, and infrastructure. The next step is focused on diagnosing the current maturity level of the organisation's software operation. This information forms the basis for focused improvement projects in the next step. Each project creates new or improved software processes, which are verified and implemented within the organisation to improve engineering practices. The final phase is focused on learning lessons from the SPI efforts (McFeeley 1996, Zahran 1998).

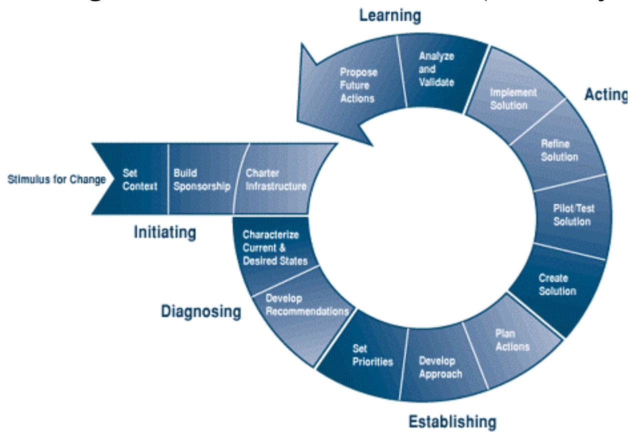


Figure 1. The IDEAL model for SPI (McFeeley 1996)

SPI initiatives typically use normative models to assess current software practices and provide guidance for how to prioritise improvements. The most popular model is the Capability Maturity Model (CMM), see Figure 2 (Humphrey 1989, Paulk *et al.* 1993). The CMM describes five levels of maturity expressed through a set of Key Process Areas (KPAs) that should be in place for each level. At the initial level, software projects proceed in an ad hoc fashion and depend on the technical skills and heroic efforts of individuals. At the repeatable level, project management disciplines are established to enhance each project's ability to set and meet reasonable time and budget commitments. At the defined level, organisation wide processes are used throughout the organisation and tailored to individual projects. At the managed level, software processes are monitored quantitatively and adjusted to better meet product quality goals. Finally, at the optimising level, quantitative data are used to continuously improve the organisation's processes (Goldenson and Herbsleb 1995, Hayes and Zubrow 1995).

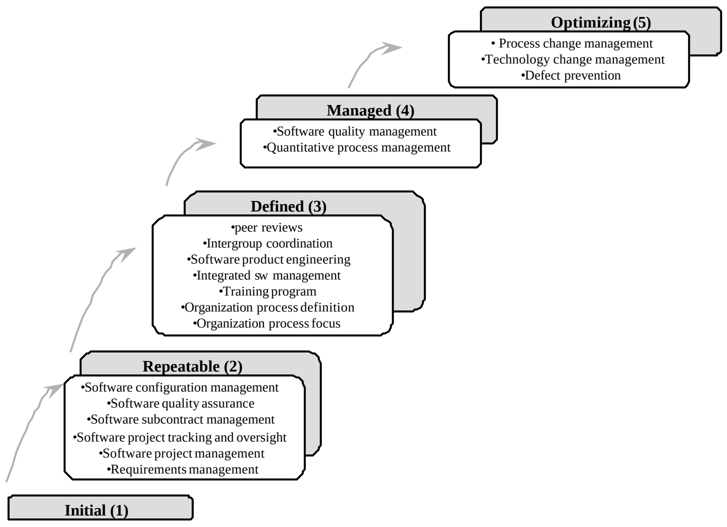

Optimizing (5)
• Process change management
•Technology change management
•Defect prevention
Managed (4)
•Software quality management
•Quantitative process management
Defined (3)
•peer reviews
•Intergroup coordination
•Software product engineering
•Integrated sw management
•Training program
•Organization process definition
•Organization process focus
Repeatable (2)
•Software configuration management
•Software quality assurance
•Software subcontract management
•Software project tracking and oversight
•Software project management
•Requirements management
Initial (1)

Figure 2. The five Maturity Levels in CMM (Paulk *et al.* 1993)

The IDEAL model and the CMM capture some of the general approaches to SPI. In practice there is plenty of room for interpretation of the models and for supplementing with other types of knowledge (Mathiassen *et al.* 2001). A discussion of the characteristic features of SPI practices is presented in (Aaen *et al.* 2001).

1.2 Knowledge Management

Software engineering is a highly knowledge intensive activity and software organisations constantly need to adopt new technologies and improve their practices. KM has therefore been used to inform practices within software organisations in general and SPI initiatives in particular. Arent and Nørbjerg (2000) analysed how organisational knowledge creation processes can support SPI initiatives. Baskerville and Pries-Heje (1999) investigated KM as a source for developing supplementary key process areas to the CMM in small and medium sized companies. Kautz and Thaysen (2001) studied how knowledge, learning and IT support occur in small software organisations. Kautz and Nielsen (2001) analysed the role of knowledge transfer in SPI implementation and developed a framework to support implementation practice. Arent *et al.* (2001) studied SPI as an organisational learning process and suggested two main strategies for learning: 1) the exploration strategy focused on knowledge sharing and learning by doing, and 2) the exploitation strategy aiming to create explicit knowledge in the form of standard processes and guidelines. These studies demonstrate the relevance and usefulness of applying KM to software organisations. It is, however, far from clear how software organisations can take advantage of KM insights on a practical level and, as part of that, develop a KM strategy to guide their SPI efforts.

Some KM approaches are based on the distinction between explicit and tacit knowledge, e.g. (Nonaka and Takeuchi 1995). The very idea in software engineering is to explicate knowledge in the form of programs to be executed on computers and software engineers spend great efforts specifying programs and models while at the same time participating in close people-to-people interactions as members of software teams. Explicit knowledge related to SPI is of two types: 1) Software engineering knowledge, e.g. about software quality assurance or software configuration management, and 2) SPI-knowledge, e.g. the IDEAL model and the CMM (Pourkomeylian 2001a). Tacit knowledge is personal, it is embedded in individual experience, and it involves intangible factors such as personal beliefs, perspectives, and underlying values (Nonaka and Takeuchi 1995). Tacit knowledge plays a major role in SPI because a deep appreciation of software practice is needed to assess present capabilities, to design useful new processes, and to implement these processes as part of the software operation. While the tacit-explicit dichotomy provides valuable understanding of SPI issues (Arent and Nørbjerg 2000, Arent *et al.* 2001, Pourkomeylian 2001a) it has certain weaknesses when it comes to strategically positioning KM within specific SPI initiatives. Most importantly, the key idea in SPI is not merely to explicate knowledge. SPI initiatives rest on the ambitious idea of creating and sharing knowledge on an organisational level across different individuals, projects, and departments. The emphasis in SPI is hence on the creation of generic, codified knowledge and on the tailoring of this knowledge to specific projects.

Consequently, we explore the strategic influence of KM on SPI practices based on a different, fundamental distinction within KM. Some approaches focus on capturing, distributing, and effectively using codified knowledge across organisations, while others emphasise creation, acquisition, and transfer of personal knowledge as part of modifying organisational behaviour (Shariq 1999, Scarbrough and Swan 1999, Zack 1999). The cognitive-based model relies on codification of knowledge objects, which are stored in databases, from where they can be accessed and used by anyone in the organisation (Swan *et al.* 1999). This model corresponds to what Hansen *et al.* (1999) call the codification approach to KM. In contrast, the community-based model provides a perspective in which knowledge is closely tied to communities of practice (Brown and Duguid 1991) and knowledge is mainly shared through direct person-to-person interaction (Swan *et al.* 1999). This model corresponds to what is called the personalisation approach to KM (Hansen *et al.* 1999).

1.3 Research Approach

This paper is one of the results of a collaborative practice research project (Mathiassen 2000) carried out in the software organisation at AstraZeneca, Mölndal, Sweden. An SPI effort has been performed from the initiating phase to implementing new, organisation wide software processes. The present study is focused on the issue of KM within this SPI project and aims to find answers to the following questions: 1) Which KM practices will support continued SPI efforts at AstraZeneca? 2) How can organisations in general develop a KM strategy to support their SPI activities?

The paper presents a case study in which a theoretical framework from KM (based on Hansen *et al.* 1999 and Swan *et al.* 1999) is used to evaluate current KM practices and to provide guidance for improving SPI practices. Data about current SPI practices were collected from a number of sources: SPI assessments, SPI schedules, the SPI implementation plan, previous research on the SPI initiative (Pourkomeylian 2001a, 2001b, 2001c, Pourkomeylian *et al.* 2001) and direct participation in the organisation's SPI efforts. These sources were systematically interpreted and evaluated using the KM framework and the result of this analysis was combined with the stated objectives of the SPI effort at AstraZeneca to develop an improved approach to KM. The KM framework and the experiences from the case study were then used to review conventional SPI ideas as presented in the literature. These more general findings were finally combined with the insights from the specific case to propose guidance on how to facilitate KM in future SPI initiatives.

Section 2 presents the adopted KM frameworks, and Section 3 presents the organisational context of the case. The ongoing SPI initiative is described and evaluated from a KM point of view in Section 4, and a KM strategy is proposed to support further SPI activities. Section 5 discusses the practical implementation of this KM strategy. Finally, Section 6 concludes the argument by discussing KM in relation to SPI in general.

2 KNOWLEDGE MANAGEMENT FRAMEWORK

The codification or the cognitive KM model focuses on ways to codify experiences, to store the resulting knowledge objects, and to reuse them through the use of Information Technology (IT) (Hansen *et al.* 1999, Swan *et al.* 1999). Knowledge is extracted from the persons who originally created it, made independent of those persons, and reused for various purposes. The computer is seen as a facility for storing and sharing knowledge objects through individual-to-electronic interactions. This allows many people to search for and retrieve the same knowledge without having to contact the person who originally developed it. IT and the so-called knowledge agents who are responsible for codifying and storing documents are seen as critical success factors.

In contrast, the personalisation or community KM model sees knowledge as socially constructed (Hansen *et al.* 1999, Swan *et al.* 1999). Knowledge is intrinsically related to individual behaviour and it varies between different communities of practice. The focus is on interaction and dialog between individuals. Knowledge is created and transferred through collaboration, in brain storming sessions, and through person-to-person conversations. IT can be used as a medium for collaboration and knowledge sharing, but collaboration and trust across different groups of actors are seen as the critical success factors for KM.

An organisation's KM strategy should in general reflect its business situation and strategy (Hansen *et al.* 1999, Zack 1999). Hansen *et al.* (1999) suggest that development of a KM strategy requires answers to the following questions. First, how does the organisation create value for customers; is reuse of codified knowledge essential because the organisation is repetitively dealing with similar problems, or, do people seek advice from colleagues to deepen their understanding of the issues and to create highly customised solutions to unique problems? Second, what kind of economic model underpins the value created? The codification approach relies on the "economics of reuse" saving work, reducing communication costs, and allowing an organisation to take on more projects. The personalisation approach relies on "expert economics" focusing on collaboration between highly skilled people and on giving advice that is critically dependent on tacit knowledge. This process of sharing expert knowledge is time consuming, expensive, and slow. Third, what kind of knowledge do people need to deliver the services of the organisation? For an organisation relying on reusing knowledge objects the individual-to-electronic approach makes most sense. In such organisations people should have the profile of implementers rather than inventors. In contrast, organisations using a personalisation approach need to hire more inventors with the analytic and creative skills required to solve unique business problems (Hansen *et al.* 1999). Table 1 illustrates the two approaches.

	Codification Approach	**Personalisation Approach**
Competitive Strategy	Provide mature, relatively standardised products or services to customers	Provide new and innovative products and services to meet customers' unique needs
Economic Model	Invest once in a knowledge asset and reuse it many times	Use highly customised solutions designed to solve specific problems
Knowledge Management Strategy	Develop IT-based networks that codifies, stores, disseminates, and allows reuse of knowledge	Develop networking activities for linking people so that personal knowledge can be shared
Information Technology	Invest heavily in IT to connect people with reusable codified knowledge	Invest moderately in IT to facilitate interaction and exchange of personal knowledge
Human Resources	Hire implementers. Train people in groups and using IT-based learning. Reward people for using and contributing to document databases.	Hire problem solvers that can tolerate ambiguity. Train people through one-on-one mentoring. Reward people for sharing knowledge with others.

Table 1. Two, contrasting KM approaches (Adapted from Hansen *et al.* 1999)

Hansen *et al.* (1999) suggest more specifically that an organisation should explore the following questions to develop its KM strategy:

1. *Does the organisation offer standardised or customised services?* Organisations that offer standardised services do not vary much, if at all. A KM approach based on codification and reuse of knowledge fits these organisations. Organisations that provide customised services

need a different approach. These organisations should consider a personalisation approach to support varying customer needs.

2. *Does the organisation have mature or innovative services?* A strategy based on mature services typically benefits most from a reuse model. The processes for developing and selling such services involve well-defined and well-understood tasks and knowledge that can be codified. A strategy based on service innovation, on the other hand, is best supported by a personalisation approach. People in organisations seeking innovation need to share information that would get lost in codified form.

3. *Do the people rely on codified or personal knowledge to solve problems?* When the employees rely on codified knowledge, such as standard procedures, guidelines, or templates, to do their work, the individual-to-electronic approach makes most sense. Personal knowledge such as scientific expertise, operational know-how, and technical expertise, is difficult to communicate in writing. It is acquired through personal experience. When people needs personal knowledge to help solve problems, the person-to-person approach works best.

An organisation's answers to these questions indicate which primary (80%) and secondary (20%) KM approach to adopt (Hansen *et al.* 1999). Making a 50%-50% decision is seldom useful because the most characteristic features of the organisation need to be emphasised and supported.

Zack (1999) argues in a similar fashion that organisations should develop a KM strategy that reflects the organisation's business situation and strategy. The approach taken is, however, different. First, the organisation's KM practice is compared to its competitors. Second, an analysis is performed of the organisation's strategic gap (between what an organisation must do to compete and what it is actually doing) and the organisation's knowledge gap (between what an organisation must know and what it actually knows). Based on those insights a strategy is developed emphasising whether the organisation primarily is creator or user of knowledge, and whether the primary sources of knowledge are internal or external.

Software organisations that engage in SPI efforts focus on improving software capabilities in an evolutionary fashion based on current practices. The perspective is in most cases internal and the intention is not primarily to position the organisation in relation to competitors. The analysis suggested by Hansen *et al.* (1999) is therefore considered more appropriate in this particular domain. The two dimensions of a KM strategy suggested by Zack (1999), i.e. creator vs. user of knowledge and internal vs. external knowledge sources, relate directly to the third question above.

3 ORGANISATIONAL CONTEXT

AstraZeneca is a research-driven pharmaceutical organisation with some 10,000 R&D personnel and a US $2 billion R&D investment in 1999. The collaborative research project was initiated in 1999 within Development IS (DevIS) to improve the software operation within the organisation (Pourkomeylian 2001a). DevIS provides IT-support to clinical and pharmaceutical projects, regulatory affairs, and product strategy and licenses at AstraZeneca R&D, Mölndal.

Many regulatory authorities require that pharmaceutical companies and their software organisations comply with GXP (Good *Manufacturing* Practice, Good *Clinical* Practice, and Good *Laboratory* Practice) rules. GXP rules are the authorities' quality requirements to pharmaceutical companies for ensuring patient health, the quality of processes (e.g. clinical studies or software development) and the quality of products (e.g. tablets or software). Every software project regulated by GXP requirements should carefully apply all quality rules and be able to show by documented evidence that the software is compliant with the related GXP requirements.

The company has adopted Standard Operation Procedures (SOPs) that explicitly describe its software quality rules. In January 2001 a special unit was created called Mölndal R&D IS Quality and Compliance (Q&C) responsible to lead all SPI activities within the company and co-ordinate all quality and compliance issues at the site level. It is this SPI Unit that is the focus of this study. The SPI initiatives leading to the formation of the unit are reported in (Pourkomeylian 2001b).

The primary customers of the SPI Unit are the software engineers and managers in DevIS. They are basically engaged with software development, software maintenance and software operation activities. Software in DevIS include the programs and all related documentation.

A maturity assessment using a modified CMM assessment method showed that DevIS was by then a CMM level 1 organisation. The steering committee of the SPI initiative gave priority to creation of a template library including templates for documentation of software development activities and improvement of the following processes: 1) the software validation, 2) the software change management, and 3) the document version control. The implementation phase includes further activities in which the created processes are improved on the basis of experiences of using them in practice (Pourkomeylian 2001c). This phase has resulted in a new version of the software processes in June 2001 and an assessment evaluating the effects of the SPI project in the organisation (Pourkomeylian *et al.* 2001).

4 KNOWLEDGE MANAGEMENT PRACTICE

We proceed by describing and evaluating current KM practices related to the SPI efforts using the framework of Hansen *et al.* (1999). Based on this evaluation we discuss design of a KM strategy to support further initiatives in the SPI Unit.

4.1 Evaluation of current practice

An overview of the current KM practices in the SPI Unit is provided in Figure 3. The SPI Unit plans and performs continuous efforts to improve software processes and practices. New or improved processes are documented together with procedures, guidelines, and templates to support software development at DevIS. The SPI Unit also plans and performs other Q&C related projects, e.g. developing a new register of software for the entire company which integrates the existing information stored in different registers in various departments at AstraZeneca R&D, Mölndal.

A main challenge is to implement new processes in the entire organisation as part of the engineers' daily work. To do this the SPI Unit helps every software project to adapt new processes to the project's situation. The SPI Unit arranges customisation meetings at the start of every software project to discuss key factors that might have some impact on the project. Examples of factors are the quality requirements and the project's organisation and size. The result of such meetings is a matrix that illustrates which documents should be produced during the project's life span and who should produce, review, and approve which document. During such meetings the SPI Unit and the project team discuss if and how software validation should be performed and how change control should be performed and documented during the project's life span. After the customisation meeting the SPI Unit and the project team agree on a tailored process for producing, reviewing, and approving documents and for performing software validation and change control in that specific project. The agreement is documented and stored by the SPI Unit and a copy is sent to the project team. The SPI Unit engages in follow up meetings to evaluate the degree to which the project follows the tailored processes. The SPI Unit collects feedback from projects concerning the usability of the processes and the templates to support further improvement efforts.

Another service that the SPI Unit offers is supporting the software projects, engineers, and managers with Q&C advice. The questions are of different types and complexity, from how to use a specific template for documenting results to explanation of new rules and policies. The SPI Unit also helps software projects taking practical Q&C responsibility. A member of the SPI Unit can act as quality assurance expert engaged by the software project to review the project and the produced documentation.

When introducing new or modified software processes the SPI Unit arranges and performs training sessions for all engineers that might be affected by the processes. The feedback from the participants is documented and stored in a database. This feedback together with the feedback from the practical use of processes is used as input to new improvement activities.

Finally, the SPI Unit plans and conducts customised audits to vendors that offer services or products to AstraZeneca or are candidates to do so. This audit service is usually based on a request from a software project or a software department. The results of the audit are documented and feedback is given both to the vendor and the organisation that ordered the vendor audit.

Some of these services are of a standardised and mature type such as performing vendor audits and offering training. In vendor audit and training you perform more and less the same steps and re-apply the same material or templates in doing them. The focus and contents of both the audit and training might differ from case to case. But quite often there is only little variation. In contrast, other services are of a more customised and innovative type like offering Q&C consulting to different customers regarding different problems.

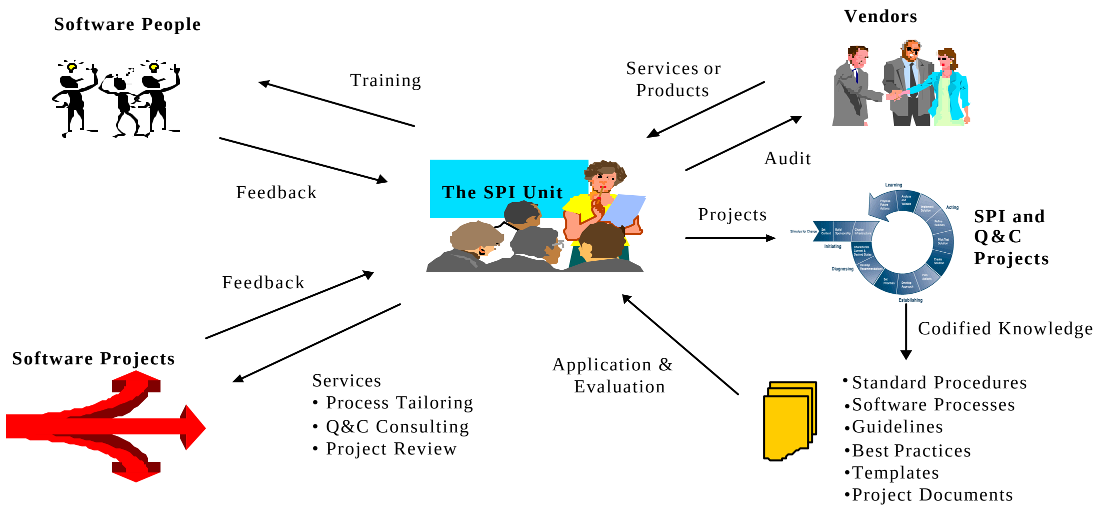


Figure 3. The SPI Unit's services

The variation in the SPI Unit's services requires access to two types of knowledge for being able to deliver high quality services to the organisation. Some of the services are primarily based on codified knowledge within different areas, such as authority requirements, standard procedures, and guidelines. Others require professionalism in form of know-how, such as planning and performing SPI activities and consulting on Q&C issues. Only rarely are the services of an either-or nature, but they each have an emphasis towards one of the extremes.

Table 2 summarises the characteristic features of the SPI Unit's services. The evaluation is based on Hansen *et al.*'s (1999) three main questions. 1) Do the organisation offer standard or customised services? 2) Are the services mature or innovative? 3) Do the practitioners need codified or personal knowledge to deliver the services to the organisation?

Service	**Standardised or Customised**	**Mature or Innovative**	**Codified or Personal**
Software Processes	Standardised (adaptable)	Innovative	P (C)
Q&C Guidelines	Standardised	Mature	C (P)
Process Tailoring	Customised	Innovative	P(C)
Q&C Consulting	Customised	Innovative	P (C)
Project Review	Standardised	Mature	C (P)
Training	Standardised (variations)	Mature	C (P)
Vendor Audit	Standardised (variations)	Mature	C (P)

Table 2. Evaluation of the SPI Unit's services

4.2 Design of KM strategy

The question is then which KM strategy the SPI Unit should adapt. As illustrated in Table 2, the SPI Unit's services differ from each other in being standardised or customised, in being mature or innovative, or in being based primarily on codified or personal knowledge. Establishing one uniform KM strategy for the SPI Unit covering all the services might therefore be unwise. Some level of differentiation between services seems appropriate. In addition, as suggested by Hansen *et al.* (1999), we should, for each group of services, adapt a main KM approach supported by a secondary approach to improve current KM practices in the SPI Unit.

A personalised approach supported by some level of codification will support the KM practises related to the following services: software processes, process tailoring, and Q&C consulting. In each of these cases standards do play a role, but the services are customised to the organisation or to the situation at hand, and they all require innovative skills and extensive use of personal knowledge. All of these services are, for these reasons, time consuming and in this sense expensive. Codified knowledge is used as a supplementary approach to provide potentially useful process models, guidelines, and templates. In this way, codified knowledge is used to inform the activities while the activity itself is enacted mainly based on personal innovative skills and know-how. There is some variation between the three services. The development of software processes is inherently dependent on personalised knowledge while there might be opportunities to further base process tailoring and Q&C consulting on codified knowledge. In considering this option it is, however, important to keep in mind that both of these activities in their current form involves close customer dialog, joint problem-solving, and knowledge sharing. The resulting feedback to the SPI Unit plays an important role in tuning the established codified knowledge base and in identifying new areas for improvement. The main KM approach for these services should therefore be personalised to support knowledge sharing and networking.

In contrast, Q&C guidelines, project review, training, and vendor audit should primarily be based on a codified KM approach supported by personal know-how as a secondary approach. These services are all of a standardised nature, even though they might lead to customised results or include certain variations. They have a more mature character than the services mentioned above. The amount of codified knowledge involved is high, but some level of personalised knowledge is still required to cope with variations. All these services require software and quality skills and they are time consuming. One way of improving the current practices is to further codify software and quality knowledge within requirement specification, software design, architectural design, and application integration.

5 FROM STRATEGY TO ACTION

The KM strategy explicates some of the tacit assumptions underpinning current SPI practices. But it also points in direction of new or modified initiatives. The SPI Unit needs to focus more on personalised approaches and it needs to explore further the interactions between codified and personalised approaches. Taking the full consequence of the KM approach is not a simple task and the SPI Unit should carefully reconsider its approach to KM in the light of its new initiatives.

Swan *et al.* (1999) distinguishes between a network model and a networking model to knowledge management and innovation. Networks based on IT are useful for sharing and storing codified knowledge whereas it is necessary to encourage and develop face-to-face networking amongst different groups and actors to share personalised knowledge. Using these terms, the KM strategy of the SPI Unit needs to rethink and improve its IT-based networks and it needs to increase its networking activities to strengthen personalised approaches to KM. In the following, we outline the SPI Unit's initiatives in response to this challenge.

5.1 Improved network facilities

The SPI Unit has now developed an Electronic Process Library (EPL) on the Intranet to increase the availability of the current base of codified knowledge. The Unit collaborated with web-experts to design the network, see Figure 4, and the SPI Unit now manages it. The EPL is continuously updated and developed to serve as a shared, centralised medium amongst software engineers, software managers, and SPI experts. In addition to processes, guidelines and templates this network includes the current SPI project specification, plan, and implementation report as well as the latest CMM assessment report. The emphasis is on codified knowledge, but the network has also been developed to include facilities to support sharing of personal knowledge.

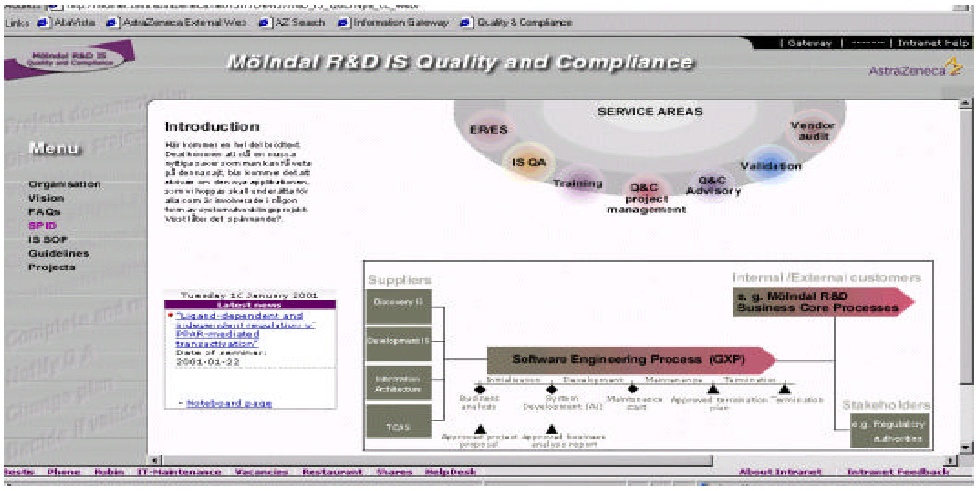

Links: AltaVista AstraZeneca External Web AZ Search Information Gateway Quality & Compliance
| Gateway | ------- | Intranet Help
Mölndal R&D IS Quality and Compliance
AstraZeneca
Menu
Organisation
Vision
FAQs
SPID
IS SOF
Guidelines
Projects
Introduction
Tuesday 16 January 2001
Latest news
Noteboard page
SERVICE AREAS
ER/ES
IS QA
Training
Q&C project management
Q&C Advisory
Validation
Vendor audit
Suppliers
Discovery IS
Development IS
Information Architecture
TCRS
Internal /External customers
e. g. Mölndal R&D Business Core Processes
Software Engineering Process (GXP)
Initialisation
Development
Maintenance
Termination
Stakeholders
e.g. Regulatory Authorities
Results Phone Robin IT-Maintenance Vacancies Restaurant Shares HelpDesk
About Intranet
Intranet Feedback

Figure 4. The SPI Unit's Electronic Process Library

Electronic search facilities have been developed to support easy and personalised access to the codified knowledge base. This is supplemented with a personalised approach where the Intranet can be used to find SPI experts or software engineers with relevant experiences. Once contacts are made, person-to-person meetings can be arranged to organise joint problem solving, to transfer experiences, or to provide advice.

Offering advice on processes and procedures in the organisation is a daily challenge for the SPI Unit. Often, the same advice is offered to different projects for solving similar problems. Therefore a FAQ (Frequently Ask Question) system has been made available on the network to make the most common questions and answers available to all practitioners. This facility is updated dynamically to reflect changes in codified knowledge and experienced problems.

Finally, the EPL is to be expanded to support creation and exchange of personalised knowledge amongst software engineers and SPI experts interested in quality and compliance issues and new software technologies. As part of this, state-of-the-art papers and reports on these and other issues will be made available to stimulate discussions and learning.

5.2 Improved networking activities

The KM strategy suggests that the major weakness of current SPI practices at AstraZeneca is the SPI Unit's insufficient focus on networking activities to support sharing of personalised knowledge amongst software engineers, software managers, and SPI experts. To further strengthen such practices the Unit decided to review the literature on networking (Augier and Vendelo 1999, Swan *et al.* 1999, Davenport *et al.* 1998, Dixon 1998, Seufert *et al.* 1999, Alavi 1999) and to evaluate networking experiences within other areas at AstraZeneca. The key findings from these activities are documented in detail in (Pourkomeylian *et al.* 2001).

According to the literature networking activities support knowledge sharing within some area of interest, they operate on both personal and organisational levels, they basically rely on informal structures, and they are activated by a need. Formal structures such as an electronic team room or an Intranet that communicates common values and ideas throughout the organisation are often needed to support the efforts. To facilitate innovation, an organisation needs both internal and external networking activities.

Seven networking activities (called competence networks) were recently established at AstraZeneca to facilitate knowledge sharing. These networks are open to all practitioners. Each network focuses on a specific area such as: Internet/Intranet technology, project management, Total Quality Management, and human computer interaction. We interviewed people working with these networks to understand how they work and why some are successful and others fail. We learned that the existing networking activities typically are informal, they are internal, and different IT-based solutions are used to facilitate interaction. A dedicated network co-ordinator, a strong commitment to participation, a critical mass of participants, an explicit purpose for each network, management support, and a balance between junior and senior participants are seen as key success factors by the participants (Pourkomeylian *et al.* 2001). The SPI Unit has used these insights and experiences to enhance and further develop its networking activities.

First, a number of structured networking activities are in place. These are mostly organised as formal meetings aiming to solve problems related to software engineering in specific projects or to share knowledge and experience gained in different software projects. An example is the customisation meetings in which different software engineering issues, such as how to validate software and how to design test facilities for testing, are discussed between the SPI Unit and members of a software project. The results of this kind of meeting are joint solutions of a specific problem in a specific project. Another example is SPI workshops at which the SPI Unit and other practitioners create new or improve existing software processes through dialog and knowledge sharing. To improve an existing process or create a new process, the SPI Unit invites different people to SPI workshops to discuss problems, experiences, and possible approaches in a specific area. The focus of these meetings is on collaborative creation of knowledge. Input should come from the experiences of the SPI Unit's members, experiences of the practitioners, codified knowledge stored in the EPL, and state-of-the-art knowledge about software engineering.

Second, less structured networking activities without formal co-ordination efforts are now encouraged and facilitated by the SPI Unit. These networking efforts are activated on an ad-hoc basis by a need for sharing knowledge and experience. An example is an internal network in the SPI Unit through which the members of the unit share knowledge and experience about different software issues. Another example is external networking activities in which members of the SPI Unit share knowledge with other SPI experts and researchers via conferences, e-mail, or telephone. Individuals use these networks when they need inspiration or information in a specific area. Input is the participant's experiences and knowledge, state-of-the-art knowledge on software engineering, and in some cases information in the EPL.

The SPI Unit must continue to develop its networking activities for several reasons. This approach supports implementation of processes and procedures, it is crucial in maintaining a strong service orientation towards the software engineers and managers, and it is useful in creating the dialog and exchange needed to prioritise future improvement initiatives. The SPI Unit should also consider to expand its horizon and engage more actively in company wide networking. This can ensure a satisfactory and uniform Q&C level globally and it can stimulate exchange and learning across various sites within AstraZeneca.

6 DISCUSSION

We have analysed the SPI initiative at AstraZeneca, Mölndal in the light of theories of KM (Hansen *et al.* 1999, Swan *et al.* 1999). We should be cautious when attempting to generalise the experiences, because SPI initiatives and software organisations are different, and because the SPI initiative at AstraZeneca has reached a certain level of maturity. We can, however, increase the usefulness of our findings by positioning them in a broader research and practice context. This final section will therefore evaluate conventional approaches to SPI, suggest lessons to other software organisations on the design of KM strategies for SPI, and review the applied KM theories in the light of our experiences.

The IDEAL model (McFeeley 1996, see Figure 1) and the CMM (Paulk *et al.* 1993, see Figure 2) are expressions of KM strategies. KM is not explicated as part of the models, but software engineering is a knowledge intensive activity and KM is implicitly integrated into SPI ideas and practices (Mathiassen *et al.* 2001).

First, the IDEAL model offers a continuous, evolutionary learning cycle for diagnosing current software practices, prioritising improvement actions, acting, and learning. This approach is primarily based on a personalised approach where the uniqueness of the software organisation is in focus, where commitments in social networks is the driving force, where each improvement project has to develop customised solutions, and where extensive practitioner participation is recommended. Codified knowledge plays a minor role in this model.

Second, the CMM offers an abstract, idealised model of a software organisation and its evolution as it matures. The model is codified knowledge of software practices presented and described on different levels of detail, i.e. five levels, specific key process areas on each level, and goals and practices for each key process area. The idea is to reuse this knowledge across software organisations to assess capabilities and to guide action. Personalised knowledge plays no role in the model itself.

Third, to reach level 2 (repeatable) software organisations are advised to implement specific key process areas (requirement management, project planning, project tracking and oversight, subcontract management, quality assurance, and configuration management). The processes should, however, not be implemented in a standardised fashion. It is recommended that each software project finds its own way to reach the goals expressed in CMM. The approach is in this phase therefore mainly personalised supplemented with some codified knowledge from CMM and state-of-the-art software engineering knowledge.

Finally, the focus changes from experiments on the project level to standardisation of processes across the software organisation when the SPI efforts continue towards level 3 (defined). Processes are explicitly defined and maintained and new processes are created to support the adaptation of the codified knowledge base to specific projects. As we approach level 3 the main KM approach becomes codified and the customisation of this knowledge to specific software projects is itself codified as a separate level 3 process.

This general KM strategy, that underpins SPI, can be combined with specific experiences from the SPI Unit at AstraZeneca to arrive at a number of lessons for how software organisations can use KM insights to guide their SPI efforts:

- *A KM approach should be defined early in the SPI project.* Software organisations are knowledge-intensive in nature and SPI initiatives are particular forms of knowledge creation, sharing, and management. The conventional approaches to SPI builds on standardised KM strategies that do not necessarily fit the needs of each individual organisation. Hansen *et al.*'s (1999) framework can, as demonstrated by the presented case, be used to diagnose the characteristics of each individual initiative and on that basis form a dedicated KM approach. This approach should reflect the emerging SPI organisation's mission and needs (Hansen *et al.* 1999, Zack 1999).

- *A KM strategy for SPI should include both codified and personalised approaches.* Both our specific experiences and the SPI literature indicate that combined approaches are needed. The question is how. Hansen *et al.* (1999) propose a 80%-20% approach. We have found that one also needs to consider differences between the services provided by an SPI unit and design specialised strategies based on the characteristic features of the services. Looking at the different nature of the IDEAL model and the CMM we suggest that improvement and innovation oriented services should mainly be based on personalised approaches whereas maintenance and diffusion of processes, procedures, and templates are more open to approaches that are mainly codified.

- *The KM strategy of an SPI initiative should change as the software organisation matures.* The difference in the underlying KM approach of CMM between level 2 and level 3 emphasises the dynamics involved in KM strategies for SPI. On a general level, the idea is to mature the software organisation's processes. It is therefore fair to say that technologies like CMM generate a push towards a codified approach. The nature of SPI is, however, learning and change. Practising SPI requires innovation and creativity despite all the codified support provided through various models. This fundamental nature of any improvement or change initiative generates a fundamental pull towards a personalised approach.

Our study has confirmed that it is relevant, and quite useful, to supplement SPI knowledge with insights from KM (Arent and Nørbjerg 2000, Baskerville and Pries-Heje 1999, Kautz and Thaysen 2001, Kautz and Nielsen 2001, Arent *et al.* 2001). Our emphasis has been on the issue of KM strategy. We have found the distinction between the codified and the personalised approaches well suited to understand the problems and challenges involved in managing knowledge in software organisations and it has helped us identify key challenges in supporting innovation at AstraZeneca.

Our experiences suggest that the key strengths of Hansen *et al.*'s framework (1999) are its focus on the issue of KM strategy, its simple and useful analytical approach to design a strategy, and its emphasis on a mixed KM strategy linked to the organisation's situation and overall business strategy. We found the key weakness of the framework to be insufficient guidance on how to operationalise and implement a specific KM strategy.

Swan *et al.*'s framework (1999) builds on an elaborate and practical view of innovation (Robertson *et al.* 1996). Technology diffusion, e.g. an SPI effort, is seen as a highly interactive and partly unpredictable process consisting of a series of episodes in which different groups iterate decisions related to the technology. These episodes include: 1) agenda formation driven by the particular needs of different groups; 2) selection of approaches based on a range of alternative options; 3) implementation supported by organisation wide adoption and training activities; and 4) usage of the technology tailored to the specific circumstances of different groups. We have found that this view describes SPI practices well, it helps maintain focus on the importance of networking activities, and it supports efforts to turn KM strategies into action. The weakness of the framework in relation to SPI and software organisations is that it tends to see the cognitive-based approach and the community-based approach as alternatives arguing for the latter and contrasting it to the former.

This leaves us with the main conclusion. SPI is a particular form of knowledge creation, sharing, and management and it is advisable to explicitly address the underlying KM strategy. Each software organisation has to find its own balance between personalised and codified approaches, this balance needs to be dynamically adjusted as the organisation matures, and the KM strategy should differentiate between different types of SPI services. The frameworks of Hansen *et al.* (1999) and Swan *et al.* (1999) complement each other well in providing guidance to this process. They build on the same basic approach to KM and they compensate for each others weaknesses.

ACKNOWLEDGEMENT

This research was sponsored by AstraZeneca R&D, Mölndal, Sweden in collaboration with the Viktoria Institute, Gothenburg, Sweden and the Institute of Computer Science, Aalborg, Denmark.

REFERENCES

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001). A Conceptual MAP of Software Process Improvement. *Scandinavian Journal of Information Systems,* Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 123-146.

Alavi, M. (1999). Knowledge Management and Knowledge Management Systems. *Journal of AIS,* Vol. 1, No.1, pp. 1-35.

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science.*

Arent, J., Pedersen, M. H. and Nørbjerg, J. (2001). Strategies for Organizational Learning in SPI. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice,* Addison-Wesley.

Augier, M. and Vendelo, M. T. (1999). Networks, Cognition and Management of Tacit Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 252-261.

Baskerville, R. and Pries-Heje, J. (1999). Managing Knowledge Capability and Maturity. In: Larsen, T., J., Levine, L., and DeGross, J., I. (Eds.): *Information Systems: Current Issues and Future Change.* Norwell, MA: IFIP/Kluwer Academic Publisher.

Brodman, J. G. and Johnson, D. L. (1995). Return on Investment (ROI) from Software Process Improvement as Measured by US Industry. *Software Process Improvement and Practice,* Vol. 1 No. 1, pp. 35-47.

Brown, J. S. and Duguid, P. (1991). Organisational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning and Innovation. *Organization Science,* Vol. 2, No. 1, pp. 40-57.

Davenport, T. H., De Long, D. W. and Beers, M. C. (1998). Successful Knowledge Management Projects. *Sloan Management Review*, Winter, pp. 43-57.

Dixon, N. M. (1998). The Responsibilities of Members in an Organisation that is Learning. *The Learning Organisation*, Vol. 5, No. 4, pp. 161-162.

Debou, C. (1997). SPI Success Factors: Toward More Business Orientation. *Software Process Newsletter, IEEE Computer Society*, No. 10, pp. 15-18.

Goldenson, D. R. and Herbsleb, J. D. (1995). *After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-009.

Haley, T. J. (1996). Software Process Improvement at Raytheon. *IEEE Software*, Vol. 13, No. 6, pp. 33-41.

Halloran, P. (1999). Organizational Learning from the Perspective of a Software Process Assessment and Improvement Program. *Proceedings of Hawaii International Conference on Systems Science*, Los Alamitos, CA: IEEE Computer Society Press.

Hansen, T., Morten, N. and Tierney, T. (1999). What's Your Strategy for Managing Knowledge? *Harvard Business Review*, pp. 106-116.

Hayes, W. and Zubrow, D. (1995). *Moving On Up: Data and Experience Doing CMM-Based Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-95-TR-008.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J. and Zubrow, D. (1994). *Benefits of CMM-Based Software Process Improvement: Initial Results.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-94-TR-13.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk, M. C. (1997). Software Quality and the Capability Maturity Model. *Communications of the ACM*, Vol. 40, No. 6, pp. 30-40.

Humphrey, W. (1989). *Managing the Software Processes.* Addison-Wesley.

Humphrey, W. S., Snyder, T. R. and Willis, R. R. (1991). Software Process Improvement at Hughes Aircraft. *IEE Software*, Vol. 8, No. 4, pp. 11-23.

Kautz, K. and Nielsen, P. A. (2001). Knowing and Implementing SPI. In: Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.): *Improving Software Organisations - From Principles to Practice*, Addison-Wesley.

Kautz, K. and Thaysen, K. (2001). Knowledge, Learning and IT Support in a Small Software Company. *Proceedings of the European Conference on Information Systems*, Bled, Slovenia.

Larsen, E. Å. and Kautz, K. (1997). Quality Assurance and Software Process Improvement in Norway. *Software Process - Improvement and Practice*, Vol. 3, No. 2, pp. 71-86.

Mashiko, Y. and Basiili, V. R. (1997). Using the GQM Paradigm to Investigate Influential Factors for Software Process Improvement. *Journal of Systems and Software*, Vol. 36, pp. 17-32.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

Mathiassen, L., Pries-Heje, J. and Ngwenyama, O. (Eds.) (2001). *Improving Software Organisations - From Principles to Practice.* Addison-Wesley.

McFeeley, B. (1996). *IDEAL. A User's Guide for Software Process Improvement.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Handbook CMU/SEI-96-HB-001.

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

Paulk, M., Curtis, B., Chrissis, M. and Weber, C. (1993). *Capability Maturity Model for Software Version 1.1.* The Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Technical Report CMU/SEI-96-TR-24.

Pourkomeylian, P. (2001a). Knowledge Creation in Improving a Software Organization. *Proceedings from IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Pourkomeylian, P. (2001b). Analysing an SPI Project with the MAP Framework. *Scandinavian Journal of Information Systems*, Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 147-165.

Pourkomeylian, P. (2001c). An Approach to Institutionalisation of Software Processes. *Proceedings from Tenth International Conference on Information Systems Development*, London, England.

Pourkomeylian, P., Hörnell, J. and Söderberg, S. (2001). Knowledge Sharing in a Software Process Improvement Unit. *Proceedings from The Second European Conference on Knowledge Management*, Bled, Slovenian.

Robertson, M., Swan, J. and Newel, S. (1996). The Role of Networks in the Diffusion of Technological Innovation. *Journal of Management Studies*, Vol. 33, No. 3, pp. 333-359.

Scarbrough, H. and Swan, J. (1999). Knowledge Management and the Management Fashion Perspective. *Proceedings from, The British Academy of Management Conference on Managing Diversity,* Manchester, England.

Seufert, A., Bach, A. and Von Krogh, G. (1999). Towards Knowledge Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 183-185.

Shariq, S. Z. (1999). How does Knowledge Transform as it is Transferred? Speculations on the Possibility of a Cognitive Theory of Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 243-251.

Swan, J., Newell, S., Scarbrough, H. and Hislop, D. (1999). Knowledge Management and Innovation: Networks and Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 262-275.

Stelzer, D., Mellis, W. and Herzwurm, G. (1998). Technology Diffusion in Software Development Processes: The contribution of Organizational Learning to Software Process Improvement. In Larsen, T. and McGuire, E. (Eds.): *Information Systems Innovation and Diffusion: Issues and Directions*, Idea Group Publisher.

Zack, M. H. (1999). Developing a Knowledge Strategy. *California Management Review*, Vol. 41, No 3, pp. 125-145.

Zahran, S. (1998). *Software Process Improvement: Practical Guidelines for Business Success.* Addison-Wesley.

KNOWLEDGE SHARING IN A SOFTWARE PROCESS IMPROVEMENT UNIT

Pouya Pourkomeylian, Jörgen Hörnell, Staffan Söderberg

Abstract: The Software Process Improvement (SPI) unit of a multinational pharmaceutical company, AstraZeneca, recently adopted a Knowledge Management (KM) strategy based on two complementary approaches, i.e. the codified and the personalised to support knowledge sharing within and outside the SPI unit. The SPI unit has developed IT-based solutions to make codified knowledge available and accessible to all practitioners in the organisation. The SPI unit has also initiated some basic networking efforts to facilitate the sharing of personalised knowledge between the relevant actors. To improve and further develop its knowledge sharing facilities the SPI unit decided to conduct a survey of the organisation's experiences with a number of competence networks that had been in operation for some years. This paper combines the results of this survey with insight from the literature on networks as facilities for knowledge sharing. On the basis of this analysis we propose that: 1) the SPI unit's business goals should be the driving forces behind the knowledge sharing approach, 2) both structured and non structured networks should be developed to support knowledge sharing within and outside the SPI unit, 3) knowledge sharing should become an integrated part of the SPI unit's daily work.

Keywords: Software Process Improvement (SPI), Knowledge Sharing, Networking.

1 INTRODUCTION

Perspectives on organisations are changing from seeing them as strictly structured and manageable systems to treating them as networks in which people are not simply data processors but also knowledge creators and innovators (Seufert *et al.* 1999, Nonaka and Kenney 1991). In such settings the focus shifts from products and systems as units of analysis to people, organisations, and the social processes that bind them together in ongoing relations (Webster 1992). Such organisations perceive knowledge creation, modification and transfer as taking place in the context of a network rather than seeing Knowledge Management (KM) merely as a process of capturing, distributing, and effectively using the most valuable knowledge (see Shariq 1999, Scarbrough and Swan 1999, Zack 1999).

Two main strategies for knowledge management has been identified: 1) the cognitive model and 2) the community model (Swan *et al.* 1999). The cognitive model relies on the codification of knowledge objects stored in databases from which they can be accessed and easily used by anyone in the organisation (Swan *et al.* 1999). This is what Hansen *et al.* (1999) call the codification strategy focusing on "codified knowledge". In contrast, the community model provides a perspective in which knowledge is closely tied to the person who creates it and is shared chiefly through direct person-to-person contacts (Swan *et al.* 1999). This is called the personalisation strategy, focusing on "personalised knowledge" (Hansen *et al.* 1999).

The most widely used contemporary approach for creating new knowledge and improving practice within the software industry is called Software Process Improvement (SPI). An SPI effort typically starts with establishing the current maturity level of the organisation and aims at improving the software processes on the basis of organisationa l goals and practitioners' competencies and experience (see Arent 2000). Recent reports have reflected upon the use of KM insights in SPI. Arent and Nørbjerg (2000) and Pourkomeylian (2001a) analysed how organisational knowledge creation processes can support SPI initiatives. Stelzer *et al.* 1998 studied how principles and technologies from organisational learning can be applied to SPI initiatives and become enablers of SPI success. Halloran (1999) investigated the relationship between an SPI approach and organisational learning. Mathiassen and Pourkomeylian (2001) explored how KM strategies can be applied to an SPI unit. Kautz and Thaysen (2001) studied how knowledge, learning and IT support occur in small software organisations. These studies indicate that KM insights are indeed useful in the context of SPI initiatives.

Within the KM there has been an increasing interest in the personalised dimension of knowledge, which is perhaps most difficult to manage as it cannot be formally communicated and is embedded in the routines and practices of the organisation (see Nonaka and Takeuchi 1995, Augier and Thanning 1999). Networks have been proposed by many researchers as facilitators for managing the creation and transformation of personalised knowledge (Seufert *et al.* 1999, Davenport *et al.* 1998, Swan *et al.* 1999, Augier *et al.* 1999). The focus of the present study is therefore on the issues of knowledge sharing for facilitating the creation, modification and transfer of personalised knowledge to find answers to: How can knowledge sharing be approached within and outside an SPI unit?

The paper is one of the results of an SPI research project carried out in a software organisation that is a part of an R&D IS organisation at AstraZeneca, Mölndal, a multinational pha rmaceutical company based in Sweden. An SPI effort was started in April 1999 and an SPI unit created in June 2000 to support the implementation of new software processes in the organisation. An earlier study done in this research project demonstrated a need to adapt and implement KM strategies for the SPI unit (see Mathiassen and Pourkomeylian 2001). To support KM efforts, the SPI unit adopted a KM strategy based on two complementary approaches, i.e. the codified and the personalised. Some IT-based solutions were developed to support fast accessibility to and availability of the codified knowledge among members of the SPI unit and other practitioners working with software engineering in the organisation. The SPI unit also initiated some basic networking effo rts to facilitate the sharing of personalised knowledge between the relevant actors.

In addition to these initiatives the organisation established some years ago seven informal structured focusing forums (called competence networks) as facilitators of knowledge sharing. These networks are open to all practitioners and their idea is to facilitate the sharing of personalised knowledge among practitioners. Each network focuses on a specific interest area such as: internet/intranet technology, project management, Total Quality Management (TQM), and Human Computer Interaction (HCI). A person responsible for co-ordinating networking activities heads each network. Some of these networks have worked well (are still working), some simply died, and others are somewhere in between.

To understand how to improve and further develop facilities for supporting the sharing of personalised knowledge within and outside the SPI unit we (the authors, all members of the SPI unit) needed to learn from the experiences of people working with these networks. We thus decided to analyse these networks to understand how they work and why some are successful and others fail. We planned and conducted interviews with members of the networks to understand the challenges in our own organisation of knowledge sharing through networking. We then combined the results of the analysis with insights on networking from the literature to propose a way to improve knowledge sharing in and outside our SPI unit.

Section 2 presents the research approach. Section 3 gives the organisational context of the case. Experience with already established network facilities is described in section 4 and networking insight in the literature is reviewed in section 5. Section 6 discusses the results of the analysis in the light of the theoretical framework presented in section 5 and proposes improvements and further development of the networking facilities for the SPI unit.

2 THE RESEARCH APPROACH

The study combines the results of a survey in which the experiments with networking at AstraZeneca, Mölndal, were analysed with insights on networking in the literature to guide the implementation of knowledge sharing facilities for the SPI unit. The paper is one of the results of a collaborative practice research project (see Mathiassen 2000). The basic approach is action research, but more traditional practice studies and experiments are applied to serve specific needs in the collaboration. The paper can be seen as a survey in which a theoretical framework for knowledge sharing is used to evaluate current knowledge sharing practices. The purpose is to provide guidance for improving the applied personalised strategy in the SPI unit (see Seufert *et al.* 1999, Swan *et al.* 1999, Augier *et al.* 1999, and Davenport *et al.* 1999).

We studied and analysed seven (of a total of seven) competence networks in the organisation that have been in use for more than three years. To gather data on the current state of knowledge sharing through these networks, we conducted qualitative interviews (see Patton 1990). To make sure that the interviews covered all aspects of interest as regards knowledge sharing we developed a structured interview guide. Our focus was on how the networks are organised, how they work, what the difficulties are in sharing knowledge through these networks, and what the success factors are in those networks that functioned well. Sixteen members (almost two persons from each network with a total 35 registered members) of these networks were interviewed (1.5-2 hours per interview). The interviewees held different roles in the networks, both as members and as network co-ordinators (a person who has co-ordinating responsibility for the network activities, such as calling members to meetings and so on). The persons were selected so as to capture the different view of how the networks operate and function.

Because we were interested in improving the networking facilities in the SPI unit, we chose to focus our study on three main areas. 1) The structure of the networks, i.e. how the networks were organised and operated. 2) The challenges, i.e. which were the difficulties of knowledge sharing through the networks. 3) The success criteria, i.e. which were the success criteria for sharing knowledge through networks. The results of the interviews were documented (by taking notes and recording the interviews on tape) and analysed in the light of the theoretical framework presented in section 5.

The findings of the analysis are used to inform further networking initiatives in the SPI unit.

3 THE CASE

AstraZeneca is a research-driven pharmaceutical company with some 10,000 R&D personnel and an R&D investment of US $2 billion in 1999. The study was planned and carried out as a part of an SPI effort in a software organisation called Development IS (DevIS) which is part of the R&D operation in Mölndal in Sweden. The aim of the SPI effort is to improve the capability of the software organisation.

The SPI project was started in April 1999 (and ended in June 2001) with the purpose of improving DevIS's software processes. The SPI effort was planned and carried out on the basis of the results of an assessment that established the current maturity level of the organisation. The steering committee of the SPI initiative gave priority to the following improvement activities given in the assessment report. 1) Establish a minimum documentation level for documenting the results of software projects and create a software documentation guideline. 2) Improve processes for software validation, software change management and document version control and create guidelines to support these processes. 3) Create a template library including templates for documentation of such software development activities as: user requirement specification, design specification, test plan, and validation plan. The implementation phase of the project included further activities in which the processes were improved on the basis of experiences of using them in practice. This phase resulted in a new version of the software process guidelines in June 2001 and an evaluation of the effects of the SPI effort in the organisation so far.

The company long ago adopted Standard Operation Procedures (SOPs) that explicitly describe the company's software quality rules. A special unit was created in January 2001 called R&D IS Quality and Compliance (the SPI unit) which was made responsible for leading all SPI activities in the company and co-ordinating all quality and compliance issues at the site level. The SPI unit consists of six persons, all with experience in software engineering, software process improvement, and quality and compliance. One of the authors is the project manager of the SPI effort and the other two are members of the SPI project team. The SPI initiatives that led to the formation of the SPI unit are reported in (Pourkomeylian 2001b).

The primary customers of the SPI unit are the employees of DevIS. They are basically engaged in software development, software maintenance, and software operation activities. The software engineering activities occur in three forms: 1) development of totally new software products (software development), 2) developing or changing existing software products (software maintenance), and 3) implementing locally software products which are globally developed or purchased. A typical software development project at DevIS is scheduled to take between six months and one year and includes analysis, design, construction, testing, and validation. Software maintenance activities consist of changes in the code or developing a completely new application for existing software products. Software products in DevIS include the software and all related documentation (e.g. user requirement specification, test plan, validation plan, validation report, and user manual).
Knowledge about the company's SOPs, software process guidelines and templates has been codified (as an Electronic Process Library (EPL)) by the SPI unit and made available and accessible to all practitioners in the company by means of IT (see Mathiassen and Pourkomeylian 2001). The SPI unit also collaborates with actors on two different levels for sharing personalised knowledge. 1) Internally within the SPI unit, individuals and small groups of individuals collaborate to share knowledge and experience of different software projects and different SPI efforts. 2) Externally the

SPI unit has established network facilities with software projects to support them in customising software processes in each software project (Customisation Meeting). Another area in focus with respect to external networking is the SPI working groups. These groups consist of people with different skills gathered together (invited by the SPI unit to SPI workshop sessions) to discuss software process problems or improve or create new software processes through dialogue and knowledge sharing. Another external networking facility through which the SPI unit shares knowledge is a global Quality and Compliance Network in the AstraZeneca group.

4 NETWORKING EXPERIENCES

Today, there are seven internal competence networks in the company that have been in operation for a number of years as facilitators for sharing personalised knowledge between practitioners. In this section we present the results of interviews with members of these network facilities to show how they function seen from the participants' point of view. The results have been divided into the three main areas: 1) the structure of these networks, 2) the challenges of knowledge sharing through networking, and 3) the criteria for implementing a successful network facility. Some of the participants' ideas have been cited below.

4.1 Networks and their structure

"A forum for people with the same interest or tasks".
"Different people with different roles or tasks within the same competence area".
"The networks can be informal if a few people are networking, but they should still be formalised to support the existence of the network. Formalisation of networks also supports the distribution of information about the ideas of networks".
"The networks should be informal (being initiated by a need), but they should have some formal meetings as connection nodes".

On the basis of a common interpretation among the practitioners interviewed, a network facility at AstraZeneca R&D Mölndal is understood as a number of people having the same area of interest and interacting with each other to be able to share knowledge and information. This understanding agrees well with what Augier *et al.* (1999) and Swan *et al.* (1999) define as a network.

The seven networks are built on a more formal basis and are divided into different competence areas, for example Lotus Notes, Internet, and Total Quality Management. Each network is co-ordinated by a network co-ordinator responsible for planning and calling network members to networking meetings. In each network the person who is more interest in networking and the network's focus area becomes the network co-ordinator. The results of each meeting are often documented and made available to all practitioners in the company. The network meetings are held on a regular basis, and the location and frequency of the meetings are most often decided by the co-ordinator of each network. Some meetings in some networks are more structured than others. Some have an agenda with different subjects that will be discussed during the meeting, and others are based mainly on open discussion with no formal agenda.

Most of the practitioners interviewed preferred the network to have an informal character while they felt that meetings should be more structured with a formal agenda for each meeting. Network members should not feel part of the bureaucracy of a formal organisation. Each individual should be able to choose whether he or she wishes to participate in any particular meeting. For this to be possible, members should know the purpose of each meeting in advance.

All seven networks had an internal focus in the company. Today there are other networks with a global focus. An example is the Quality and Compliance network, which includes different quality managers from different sites at AstraZeneca collaborating to share knowledge.

Different IT or technical solutions such as telephones, e-mail, videoconferencing and discussion forums were considered an important extension of face-to-face meetings, especially when members were in different geographic locations (see Alavi 1999, Scarbrough and Swan 1999, Scarbrough *et al.* 1999, Hansen *et al.* 1999).

4.2 Challenges of Networking

"Lack of one person who co-ordinates all network activities".
"Not having time dedicated for networking".
"Some people are not as committed as they should be, they are just listeners".

A lack of a dedicated network leader to co-ordinate meetings, assemble members and ensure that there are regular meetings to address interesting topics and a lack of time on the part of both network members and network co-ordinators made it difficult for networks to function well. Some network co-ordinators also saw a lack of commitment on the part of participants as an issue. Members of some networks were only listeners and did not participate in discussions. On the other hand some members felt that they could not put more time into networking because of their regular work. They mentioned a need for more time to be formally dedicated by their managers for networking. They felt that management should encourage networking and provide resources to support the creation and transfer of knowledge and experience in the company.

None of the networks followed a defined network strategy that addressed how networking should take place. Almost all interviewees mentioned the need of a networking strategy to support and address different issues such as organisation, participation, and knowledge sharing. The issue of adopting a network strategy has been treated by Seufert *et al.* (1999).

4.3 Critical Success Factors

"You have to show interest in networking, otherwise you get nothing back. What the network is focused on is important. It should be relevant to the interests of the participants".
"Networks have to be for everyone, free to participate. It works much better if you have a critical mass of interested people".
"It must be possible for us to participate in meetings without giving up our coffee breaks".

The ability to discuss software problems and perhaps find solutions through sharing information and knowledge was considered to be a major driving force for participating in the networks. It is important that the knowledge flowing through the network is relevant to the participants. The knowledge should contribute to their work; otherwise, it is difficult to motivate spending time on networking. Basically, the purpose of networking must be visible to those involved in the networks. The practitioners considered it important to have a network that is open to everyone with different levels of experience, i.e. a network should consist of both experienced and junior persons. This supports the creation and transfer of personalised knowledge from more experienced practitioners to junior participants.

The seven existing networks function both on an individual level (between the individuals in a network) and on a group and organisational level (between a group of individuals in the same network and/or between different networks). The networks consist of practitioners with different skills from different IS organisations at AstraZeneca R&D Mölndal (there are four local IS departments with the company).

It is important according to the interviewees, that a network is built around a critical mass of people with extensive knowledge. Otherwise, it is difficult to create and transfer relevant knowledge through the network. While the number of participants in a network is important, the common view of the respondents is that the total number should be kept within the range of 10-15 persons so that it is possible to create relations and initiate discussions at meetings.

The interviewees also identified a dedicated network leader with time and resources to co-ordinate network activities as a critical success factor for networking. Finally management support of networking efforts was emphasised. It is very important that management provides the necessary conditions for networking (such as time for networking and tools for facilitating the knowledge sharing activities).

To summarise the interviewees stand points, networks in AstraZeneca are understood as groups of people having the same area of interest interacting with each other for knowledge sharing. None of the networks are based on a networking strategy. The structure of networks is informal while most networking activities are co-ordinated in a more structured and formal way. All the seven networks operate internally in the company and different IT-based solutions are used to facilitate networking efforts. Not having a dedicated network co-ordinator and lack of commitment were identified as issues by the interviewees. According to the interviewees, success in networking depends on additional factors, such as knowing the purpose of the network and having a critical mass for networking, management commitment, and a balance between junior and senior members.

5 INSIGHT INTO NETWORKING

To learn about general insights into knowledge sharing through networks we studied the relevant literature. We decided to focus on Augier *et al.* 1999, Swan *et al.* 1999, Davenport *et* al. 1998, Dixon 1998, and Seufert *et al.* 1999 because they address our concerns related to networking, i.e. the structure of the networks, the challenges of networking, and the criteria for success in networking. This section presents insights from these source.

One way to describe networks is as media for knowledge sharing on both a personal and an organisational level (Augier *et al.* 1999 and Swan *et al.* 1999). One important aspect of networks is their structure. Does a network have a rigid structure like that of an organisation? Does it have a clear hierarchy? The answer to these questions is in most cases no, because networks represent loose couplings among the entities they include. A network is then by definition an informal structure in contrast to other more formal aspects of organisations (see Augier *et al.* 1999).

Networks can be external or internal, i.e. they can involve participants outside the organisation or can be strictly delimited by company boundaries (see Swan *et al.* 1999). Swan *et al.* (1999) suggest that, in order to be innovative (to facilitate the diffusion and adoption of new ideas), an organisation needs both internal and external networks. External networking can provide employees with alternative viewpoints and insights that might be useful for their organisation, but there must be internal networking in order to gain acceptance for and to share knowledge to add value to the company. Internal networks are not just for the distribution of new ideas that may later be transformed into organisational knowledge. They are also necessary for bringing together the different skills needed to take maximum advantage of any potential innovation and for implementing them throughout the company (see Swan *et al.* 1999).

Davenport *et al.* (1998) and Seufert *et al.* (1999) discuss an appropriate architecture or infrastructure in a network initiative. They argue that it is not only the informal network (i.e. the people connected by loose couplings) that is important, but also the more formalised parts. These are made up of formal structures, for example an electronic team room or an intranet that communicates common values and ideas. In general, it is important to make use of modern communication technologies as a support for the network (especially if the network is distributed over different time zones and physical locations).

One possible problem in knowledge sharing according to Dixon (1998) is the issue of resistance by management since the transfer of knowledge also means a transfer of power. Davenport *et al.* (1998) suggest in addition that management can be less supportive of a network initiative because network members spend too much resources networking. Dixon (1998) points out that all members must recognise their roles and responsibilities in the network in order for the network to function.

According to Seufert *et al.* (1999) three things are of particular importance in implementing a successful network: 1) Interconnect different levels and areas of knowledge. Different levels and areas of knowledge must be connected, since new knowledge is created from several sources of older knowledge. A network strategy should therefore consider different kinds of knowledge as well as different dimensions of the organisation (e.g. individuals, groups, and departments). 2) Interconnect knowledge work processes and knowledge network architectures. An appropriate architecture, or infrastructure, is also crucial. Both informal and formal structures are required since knowledge creation and transfer can occur in a physical meeting room, an electronic distributed team room, or in a “mental room”, i.e. through common values and ideas. 3) Interconnect knowledge work processes and facilitating conditions. The facilitating conditions are very important and include the organisational structure, management systems, and the corporate culture. This may imply establishing a corporate culture that does not suffer from the “not invented here” syndrome.

Summarising insight on networking in Augier *et al.* 1999, Swan *et al.* 1999, Davenport *et al.* 1998, Dixon 1998, and Seufert *et al.* 1999, one can see networks as media for knowledge sharing on both a personal and organisational level. Networks are basically informal structures and should be activated by a need. Formal structures such as an electronic team room or an intranet that communicates common values and ideas throughout the organisation are needed to support the knowledge sharing efforts. To facilitate the diffusion and adoption of new ideas, an organisation needs both internal and external networks. Successful network facilities should interconnect: 1) different levels and areas of knowledge, 2) knowledge work processes and knowledge network architectures, and 3) knowledge work processes and facilitating conditions.

6 DISCUSSION AND PROPOSAL TO KNOWLEDGE SHARING APPROACH

This section discusses networking experiences at AstraZeneca (section 4) in the light of the literature on networking (section 5) and proposes improvements to knowledge sharing facilities in the SPI unit.

6.1 Discussion

Formalisation: One possible reason why people prefer more formal meeting structures at AstraZeneca Mölndal is that management has not formally addressed networking as an appointed facility for knowledge sharing. The networks have been initiated by the participants. As there has been no "push" from management to facilitate knowledge sharing via networking the meetings must have a more formal form in order to survive.

Externalisation: The seven networks studied at AstraZeneca in Mölndal have their major focus in the company. However there are other globally organised networks at the company that have members from different AstraZeneca sites (e.g. the Quality and Compliance Network). Our understanding of why the seven networks studied do not have an external orientation is that external networks (global networks) require more active involvement of management. For each global network, there is a network leader who has been appointed by management with dedicated time to facilitate networking. The seven networks studied have no (management appointed) person who is responsible for expanding the network globally.

Purpose: What we found in the interviews was an unclear understanding from the practitioners' point of view of why a network exists. We believe that this problem occurs because of lack of a networking strategy. Adopting a networking strategy that addresses a clear purpose of the networking, means for communication and the role and responsibility of management may help to reduce this problem (see Davenport *et al.* 1998, Hanssen *et al.* 1999).

Commitment: Another challenge identified by the practitioners is a lack of commitment to networking efforts among network co-ordinators and members. This was also the most frequent answer of interviewees to why networks do not work. In our understanding, there is however a commitment from both the members and the network co-ordinators for networking. But because of lack of dedicated time for networking their capacity for participating in networking efforts is limited. We believe that a networking strategy that addresses the important issue of resources spend on networking could help to reduce this problem.

Co-ordination: One interesting finding as regards network co-ordinators is that all interviewees in our study considered a leader or co-ordinator to be absolutely critical for the network, not only for its start-up but for its continuation as well. This is somewhat contradictory to what Davenport *et al.* (1998) consider to be an indicator of success, namely that the network must be an organisational initiative and not an individual project. The seven studied networks are, however, mainly individual projects. Many networking efforts are planned and organised by the network co-ordinators. Important questions such as: how to initiate network facilities, what is the purpose of networking, how to provide time for networking, and how to relate networking to individual and company goals are not addressed within AstraZeneca Mölndal. That makes the network co-ordinator's role crucial for the establishment and survival of networks.

Resistance: We did not find any indicators of resistance by management (see Dixon 1998) to be an issue in this study. By participating in a network and sharing knowledge you might however reduce your relative power as an employee. To motivate people to participate despite this possible drawback, they must receive something in return - either the understanding that they themselves gain knowledge and thereby power or some kind of reward according to a system implemented on a company level. This again relates to the issue of motivation for participation. There must be some evidence that the network can help employees in their jobs or to develop general competencies of importance to their profession. One way to do this is for the entire company to acknowledge the value of networking and commit to the idea of knowledge sharing. This is what Seufert *et al.* (1999) and Davenport *et al.* (1998) mean when they talk about the need for facilitating conditions and a knowledge friendly culture. Our study did not find any indications that there should be major obstacles to develop such a culture further at AstraZeneca in Mölndal.

Critical mass: We encountered several opinions about the size of a network. Most interviewees said that a network should not be ”too big”. A maximum of about 10-15 persons was suggested because a larger network lacks the necessary personal touch and becomes a group of anonymous participants. On the other hand, some of the participants said that the minimum limitation for a network is two persons and that there is no upper limit. The participants’ different feelings about the size of a network are very likely related to their standpoints on the structure of the networks. Planning and organising meetings with more than 10-15 people require a great deal of planning and hence a more formal structure. In contrast, networking through e-mail, telephones, and video conferencing systems on a more ad-hoc basis does not require extensive planning.

Adding value: The interviewees saw, as mentioned, the notion of adding value as a critical success factor. Attention to both individuals and company goals is an important factor for implementing successful knowledge sharing facilities. If both the members of networks and the management understand how networking can help satisfy personal goals and business goals, it may be easier to gain commitment on all levels (see Dixon 1998 and Davenport *et al.* 1998). This issue should be addressed in any networking strategy that is adopted.

6.2 Improving knowledge sharing in the SPI unit

The networking efforts of the SPI unit should be based on the overall business goals of the organisation combined with an understanding of the needs and behaviours of the relevant individuals related to the unit. These overall concerns should be expressed in a networking strategy that should be communicated to all practitioners on different levels of the organisation. Different types of networking facilities should be developed to support the different needs of the unit’s customers and networking should be developed further as an integral part of the unit’s daily work.

Based on these insights and existing facilities for networking in relation to the SPI unit figure 2 illustrates an overall design of the networking facilities.

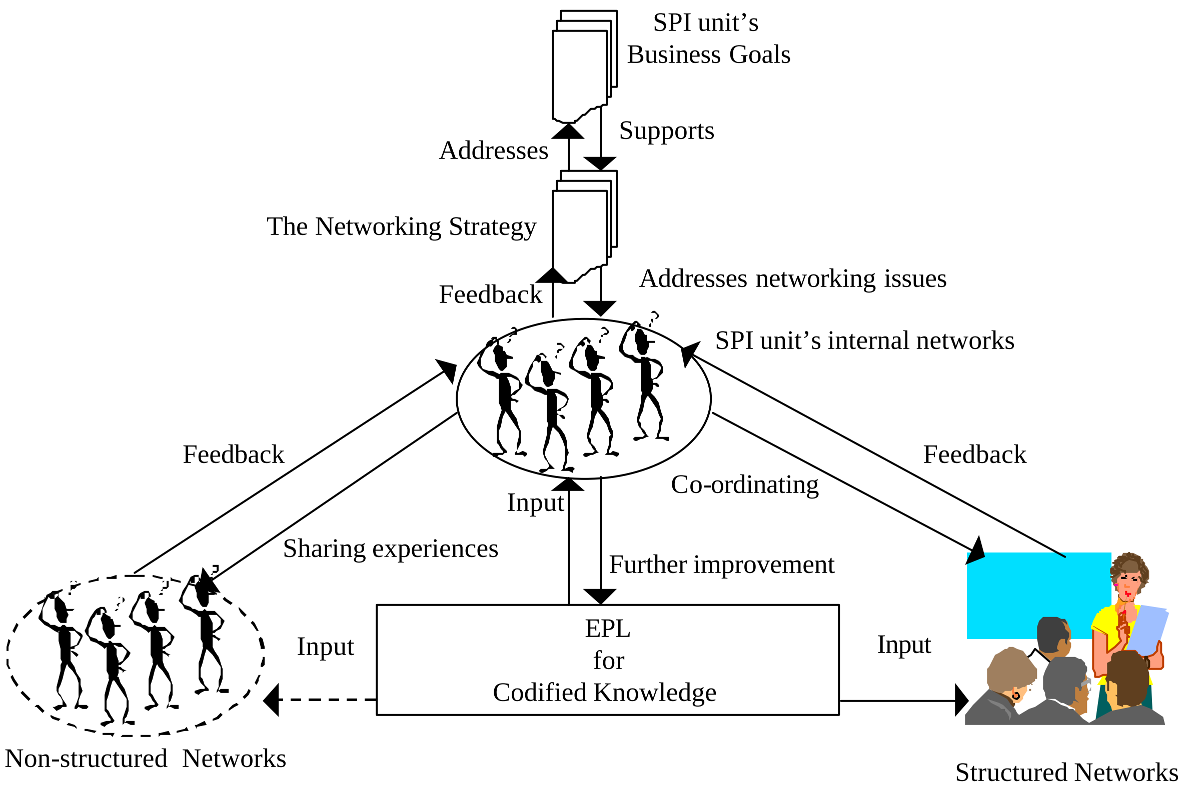


Figure 1. Proposed design of knowledge sharing facilities for the SPI unit

The main task in the SPI unit is to manage knowledge about software engineering practices at AstraZeneca. Our KM strategy combines a codified approach and a personalised approach (see Mathiassen and Pourkomeylian 2001). An IT-based facility has been implemented to support the sharing of codified knowledge within and outside the SPI unit (the Electronic Process Library, EPL). The SPI unit needs to further develop a number of structured and non-structured networking facilities to implement its KM strategy. The networking strategy should address the following issues: What types of networking facilities are needed for different customers? How should the networks be organised? How much time should the practitioners put into networking efforts? How does networking add value to practitioners and to the organisation? the networking strategy should then be communicated to all members of the SPI unit.

Structured networks are co-ordinated by the SPI unit, organised mostly as formal internal and external meetings aiming to solve problems related to software engineering in specific projects or to share knowledge and experience gained in different software projects. An example is customisation meetings in which different software engineering issues, such as how to validate software and how to design test facilities for testing, are discussed between the SPI unit and members of a software project. The results of this kind of meeting are joint solutions of a specific problem in a specific project. Another example of structured network facilities is SPI workshops at which the SPI unit and other practitioners create new or improve existing software processes through dialog and knowledge sharing. To improve an existing process or create a new process, the SPI unit invites different people to SPI workshops to discuss problems, experiences and possible approaches in a specific area. The focus of these meetings is the creation of knowledge. Input at these meetings should come from the experiences of the SPI unit's members, experiences of the practitioners, codified knowledge stored in the Electronic Process Library (EPL), and state-of-the-art knowledge about software engineering.

Non-structured networks are networking facilities without a formal organisation. Less co-ordination efforts are made in such networks. Networking efforts are activated on an ad-hoc basis by a need for sharing knowledge and experience. An example is the internal network in the SPI unit through which the members of the unit share knowledge and experience internally about different software issues. Another example is external networking activities in which the members of the SPI unit share knowledge with other practitioners via e-mail or telephone. There is less need for co-ordinating efforts in these networks and individuals use the network when they need information in a specific area. Input to non-structured networking facilities should be the participant's experiences and in some cases information in the EPL. The role of the SPI unit's members in non-structured networks should be that of participants.

The SPI unit has gained important insights from studying experiences with networking at AstraZeneca in Mölndal and the literature on networking to support sharing of personalised knowledge. In summary, these activities suggest that the SPI unit should:

- Identify further external networking facilities to support all its customers.
- Develop structures for collecting, analysing and evaluating feedback from both structured and non-structured networking facilities to improve its networking strategy.
- Internally develop structured and non-structured networks to support the creation of knowledge through the experience gained in different software projects.
- Use the EPL and other IT-based media (e.g. e-mail and intranet) to share knowledge internally and externally.

ACKNOWLEDGEMENT

This research was sponsored by AstraZeneca R&D Mölndal, Sweden in collaboration with the Viktoria Institute Gothenburg, Sweden, and the Institute of Computer Science, Aalborg, Denmark.

REFERENCES

Alavi, M. (1999). Knowledge Management and Knowledge Management Systems. *Journal of AIS*, Vol. 1, No.1, pp. 1-35.

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

Arent, J. and Nørbjerg, J. (2000). Organizational Knowledge Creation: A Multiple Case Analysis. *Proceedings of Hawaii International Conference on Systems Science.*

Augier, M. and Vendelo, M. T. (1999). Networks, Cognition and Management of Tacit Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 252-261.

Davenport, T. H., De Long, D. W. and Beers, M. C. (1998). Successful Knowledge Management Projects. *Sloan Management Review*, Winter, pp. 43-57.

Dixon, N. M. (1998). The Responsibilities of Members in an Organisation that is Learning. *The Learning Organisation,* Vol. 5, No. 4, pp. 161-162.

Halloran, P. (1999). Organizational Learning from the Perspective of a Software Process Assessment and Improvement Program. *Proceedings of Hawaii International Conference on Systems Science*, Los Alamitos, CA: IEEE Computer Society Press.

Hansen, T., Morten, N. and Tierney, T. (1999). What's Your Strategy for Managing Knowledge? *Harvard Business Review*, pp. 106-116.

Kautz, K. and Thaysen, K. (2001). Knowledge, Learning and IT Support in a Small Software Company. *Proceedings of the European Conference on Information Systems*, Bled, Slovenia.

Mathiassen, L. (2000). Collaborative Practice Research. *Proceedings of the IFIP TC 8 WG 8.2 Working Conference on Organizational and Social Perspectives on Information Technology*, Aalborg, Denmark.

Mathiassen, L. and Pourkomeylian, P. (2001). Knowledge Management in a Software Process Improvement Unit. *International Conference on Managing Knowledge: Conversations and Critiques, University of Leicester, England.*

Nonaka, I. and Kenny, M. (1991). Towards a new theory of innovation management: a case study comparing canon, Inc. and Apple Computer, Inc. *Journal of Engineering and Technology Management*, Vol.8, pp. 67-73.

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

Patton, M. Q. (1990). *Qualitative Evaluation and Research Methods*. Sage Publications.

Pourkomeylian, P. (2001a). Knowledge Creation in Improving a Software Organization. *Proceedings from IFIP WG8.6 Fourth Working Conference on Diffusing Software Product and Process Innovations*, Banff, Canada.

Pourkomeylian, P. (2001b). Analysing an SPI Project with the MAP Framework. *Scandinavian Journal of Information Systems*, Special Issue on Trends in the Research on Software Process Improvement in Scandinavia, Vol. 13, pp. 147-165.

Scarbrough, H. and Swan, J. (1999). Knowledge Management and the Management Fashion Perspective. *Proceedings from, The British Academy of Management Conference on Managing Diversity,* Manchester, England.

Scarbrough, H. and Swan, J., ed. (1999). *Case Studies in Knowledge Management*. London: Institute of Personnel and Development.

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

Seufert, A., Bach, A. and Von Krogh, G. (1999). Towards Knowledge Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 183-185.

Shariq, S. Z. (1999). How does Knowledge Transform as it is Transferred? Speculations on the Possibility of a Cognitive Theory of Knowledge. *Journal of Knowledge Management*, Vol. 3, No. 4, pp. 243-251.

Stelzer, D., Mellis, W. and Herzwurm, G. (1998). Technology Diffusion in Software Development Processes: The contribution of Organizational Learning to Software Process Improvement. In Larsen, T. and McGuire, E. (Eds.): *Information Systems Innovation and Diffusion: Issues and Directions*, Idea Group Publisher.

Swan, J., Newell, S., Scarbrough, H. and Hislop, D. (1999). Knowledge Management and Innovation: Networks and Networking. *Journal of Knowledge Management*, Vol. 3, No. 3, pp. 262-275.

Webster, F. E. (1992). The changing role of marketing in the corporation. *Journal of Marketing*, Vol. 56.

Zack, M. H. (1999). Developing a Knowledge Strategy. *California Management Review*, Vol. 41, No 3, pp. 125-145.

III INDUSTRIAL CONTRIBUTIONS

The industrial contributions of this thesis are described in a number of documents. The documents are collected in a separate volume and are available only for the committee of this thesis for reasons of confidentiality. The following gives a brief description of each document.

1. ***The Project Specification *:*** This document was created by me early 1999 and was the contract between the project and the sponsor of the project, defining the conditions under which the project should be executed.
2. ***The CMM Questionnaire *:*** This document consists of questions about how system development activities occur within DevIS at AstraZeneca in Mölndal.
3. ***The Software Process Improvement Report 1999 *:*** This document summarises the main suggestions for improvements as results of the quality improvement activities carried out at DevIS. These activities include the SPI initiatives (the CMM assessment) and other quality efforts made at DevIS.
4. ***The Software Guidelines:*** Three guidelines were created as the results of the SPID project: 1) Document Description Guideline: the purpose of this document is to describe the contents of minimum documentation requirements needed in a system development project, 2) Software Validation Guideline: the purpose of this document is to support practitioners in understanding the basics of software validation, 3) Change and Version Control Guideline: the purpose of this document is to describe the change control process in the system development and maintenance projects.
5. ***The Templates *:*** 27 templates were created to support the documenting of the results of software development activities in software projects.
6. ***The Implementation Plan:*** The purpose of this document is to describe the implementation plan for SPID improvements.
7. ***The EPL *:*** The EPL includes among others software guidelines, the process maps, SOPs, and better practices.
8. ***The Projects:*** The SPI unit supports 21 software projects (October 2001) to customise the new processes to fit their conditions within AstraZeneca in Mölndal. The projects are of different characters such as: development, change or maintenance activities, and software validation activities. A list of all projects that the unit supports is available through the EPL.
9. ***The SPID Questionnaire *:*** This document consists of questions about how the SPI unit's services supported the implementation efforts in DevIS.
10. ***The Networking Questionnaire *:*** This document consists of questions about how networking facilities work in DevIS.
11. ***The Final Project Report for SPID 2001 *:*** This document summarises the results of the SPID project and is focused on reporting the final results of the improvement activities performed within the scope of the project.

The contents of the documents with * are illustrated as follows:

The Project Specification
The following issues are addressed in this document:

- Project sponsor/System owner

- Project Manager (i.e. PGM)
- Background
- Project Deliverables
- Objectives and Expected Effects
- Conditions and Limitations
- Scope
- Dependencies
- Project Plan
- Project organisation
- Costs and Resources
- Risk Assessment and Proposed Measures
- Reuse of Existing Knowledge and Experience
- Quality Assurance
- System Classification
- Project Completion
- References

The CMM Questionnaire
The questionnaire is divided into the following sections:

1. Requirement Management (including 14 questions)
2. Software Project Planning (including 12 questions)
3. Software Project Tracking and oversight (including 16 questions)
4. Software Quality Assurance (including 12 questions)
5. Software Configuration Management (including 15 questions)
6. Conclusion (including 4 questions)

The Software Process Improvement Report 1999
The document includes the following sections:

1	Introduction
1.1	Background and purpose
1.2	Contents of Document
2	The CMM assessment
2.1	Assessment
2.1.1	Basic introduction to CMM
2.1.2	Questionnaire
2.1.3	Analysis
2.2	CRIM’s Maturity Profile
2.3	Requirement Management (Kravhantering)
2.3.1	Strengths
2.3.2	Weaknesses
2.4	Software Project Planning (Projektplanering)
2.4.1	Strengths
2.4.2	Weaknesses
2.5	Software Project Tracking and Oversight (Projektuppföljning)
2.5.1	Strengths

2.5.2 Weaknesses
2.6 Software Quality Assurance (Kvalitetssäkring)
2.6.1 Strengths
2.6.2 Weaknesses
2.7 Software Configuration Management (Konfigurationshantering)
2.7.1 Strengths
2.7.2 Weaknesses
2.8 Respondents' profile
2.8.1 Respondents' improvement priority
2.8.2 Respondents' improvement suggestions
2.8.3 Respondents' response to assessment
3 The software quality manual (SQM)
3.1 Rules
3.2 Best Practices
3.3 Models
3.4 Sources
3.5 Strengths
3.6 Weaknesses
4 Improvement suggestions
4.1 Findings
4.1.1 Change Control
4.1.2 Collect and Reuse Data and Experiences
4.1.3 Documented Processes
4.1.4 Minimum Documentation Requirement
4.1.5 Continued Development and Implementation of sqm
4.2 Action Plan

The Templates

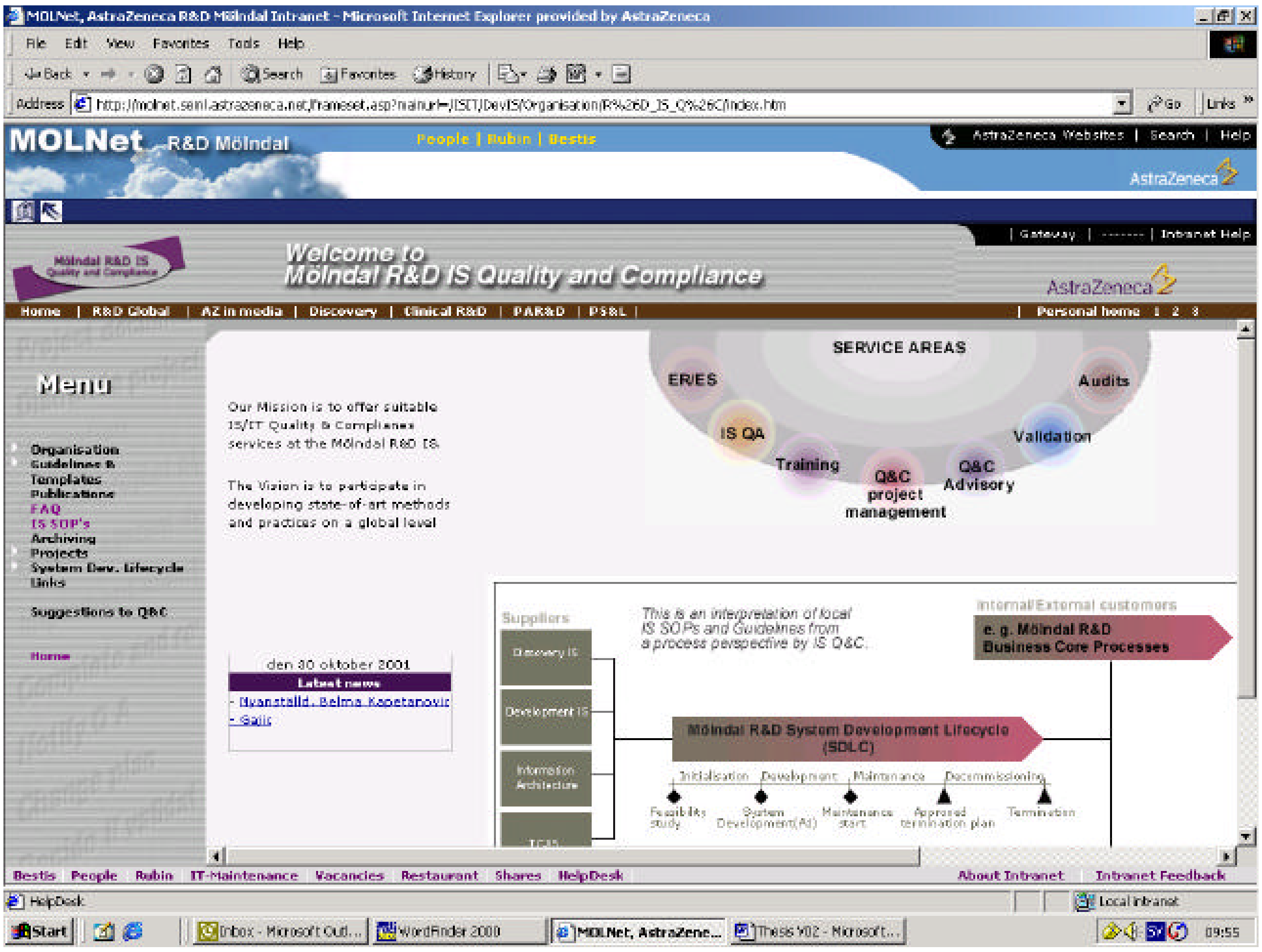
The following templates were created to support the documentation of software projects:

1. Project Specification
2. Quality Plan
3. Validation Plan
4. User Requirement Specification
5. Project Plan
6. Training Plan
7. Validation Protocol 1
8. Functional Specification
9. Installation Requirement Specification
10. System Technical Design Specification
11. Test Plan
12. User Requirement Test Specification
13. Functional Test Specification
14. Installation Test Specification
15. User requirement Test Report
16. Functional Test Report
17. Installation Test Report
18. Logistic Plan

19. Test Report
20. Validation Protocol 2
21. Maintenance Documentation
22. User Documentation
23. Change Plan
24. Validation Protocol 3
25. Project Report
26. Validation Report
27. Termination Plan

The EPL

The following are some figures illustrating the EPL:


MOLNet, AstraZeneca R&D Mölndal Intranet - Microsoft Internet Explorer provided by AstraZeneca
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://molnet.semi.astrazeneca.net/frameset.asp?mainurl=/IS/IT/DevIS/Organisation/R%26D_IS_Q%26C/index.htm
Go Links
MOLNet R&D Mölndal
People | Rubin | Bestis
AstraZeneca Websites | Search | Help
AstraZeneca
| Gateway | ------- | Intranet Help
Mölndal R&D IS Quality and Compliance
Welcome to Mölndal R&D IS Quality and Compliance
Home | R&D Global | AZ in media | Discovery | Clinical R&D | PAR&D | PS&L |
| Personal home 1 2 3
Menu
Organisation
Guidelines & Templates
Publications
FAQ
IS SOP's
Archiving
Projects
System Dev. Lifecycle
Links
Suggestions to Q&C
Home
SERVICE AREAS
ER/ES
Audits
IS QA
Validation
Training
Q&C project management
Q&C Advisory
Suppliers
Discovery IS
Development IS
Information Architecture
TCIS
Internal/External customers
e. g. Mölndal R&D Business Core Processes
Mölndal R&D System Development Lifecycle (SDLC)
Initialisation
Development
Maintenance
Decommissioning
Feasibility study
System Development(AI)
Maintenance start
Approved termination plan
Termination
Bestis People Rubin IT-Maintenance Vacancies Restaurant Shares HelpDesk
About Intranet Intranet Feedback
HelpDesk
Local intranet
Start
Inbox - Microsoft Out...
WordFinder 2000
MOLNet, AstraZene...
Thesis V02 - Microsoft...
09:55

Our Mission is to offer suitable IS/IT Quality & Compliance services at the Mölndal R&D IS.

The Vision is to participate in developing state-of-art methods and practices on a global level

den 30 oktober 2001

Latest news

- Nyanställd, Belma Kapetanovic - Gajic

This is an interpretation of local IS SOPs and Guidelines from a process perspective by IS Q&C.

Figure I. The SPI unit's home page

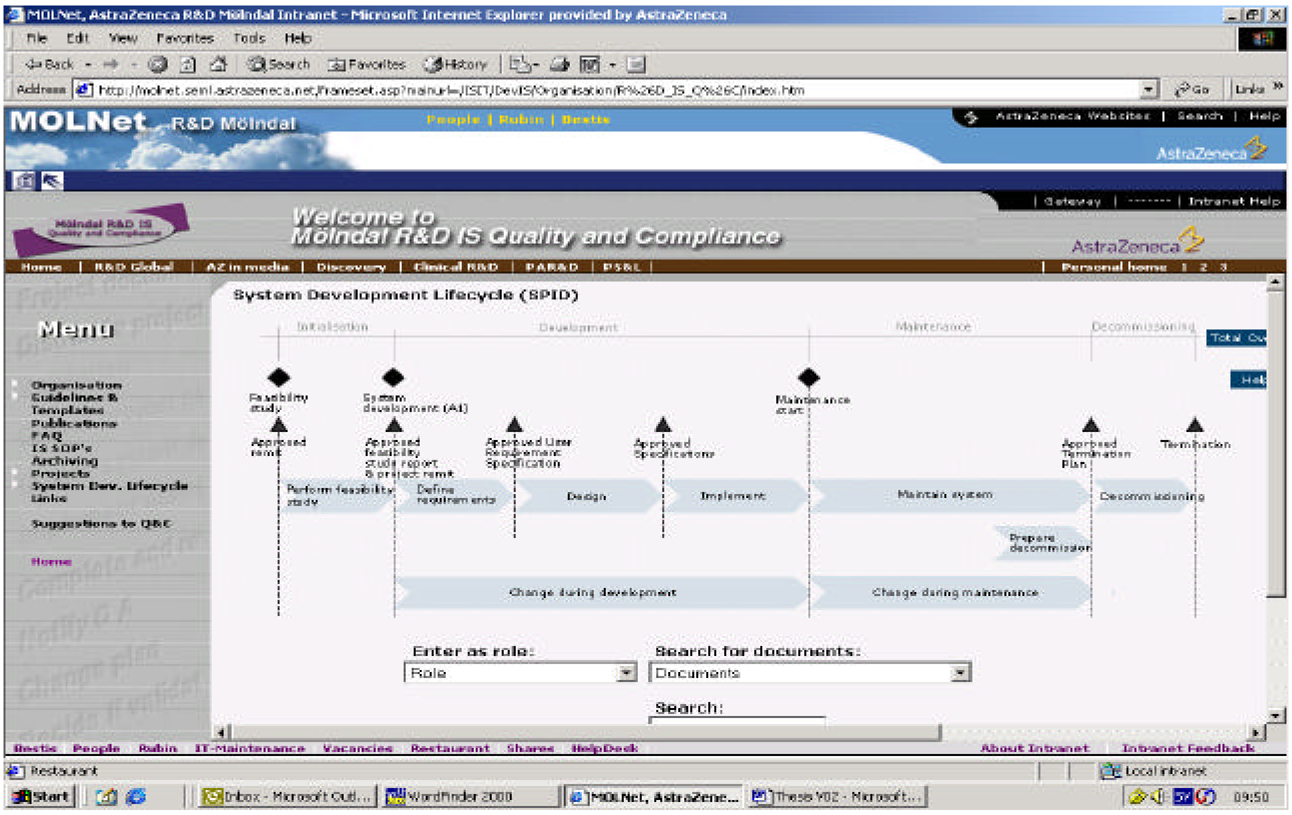


Figure II. The System Development Life cycle

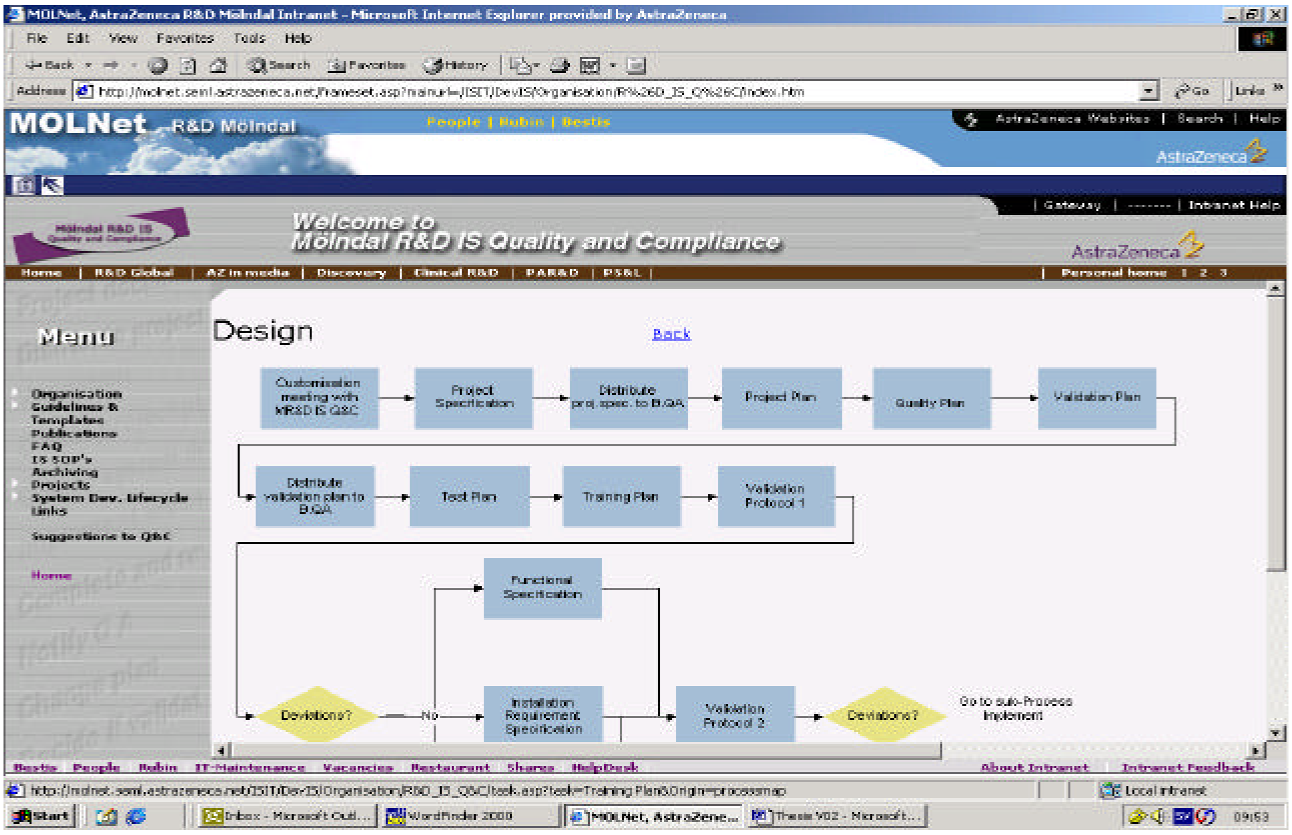
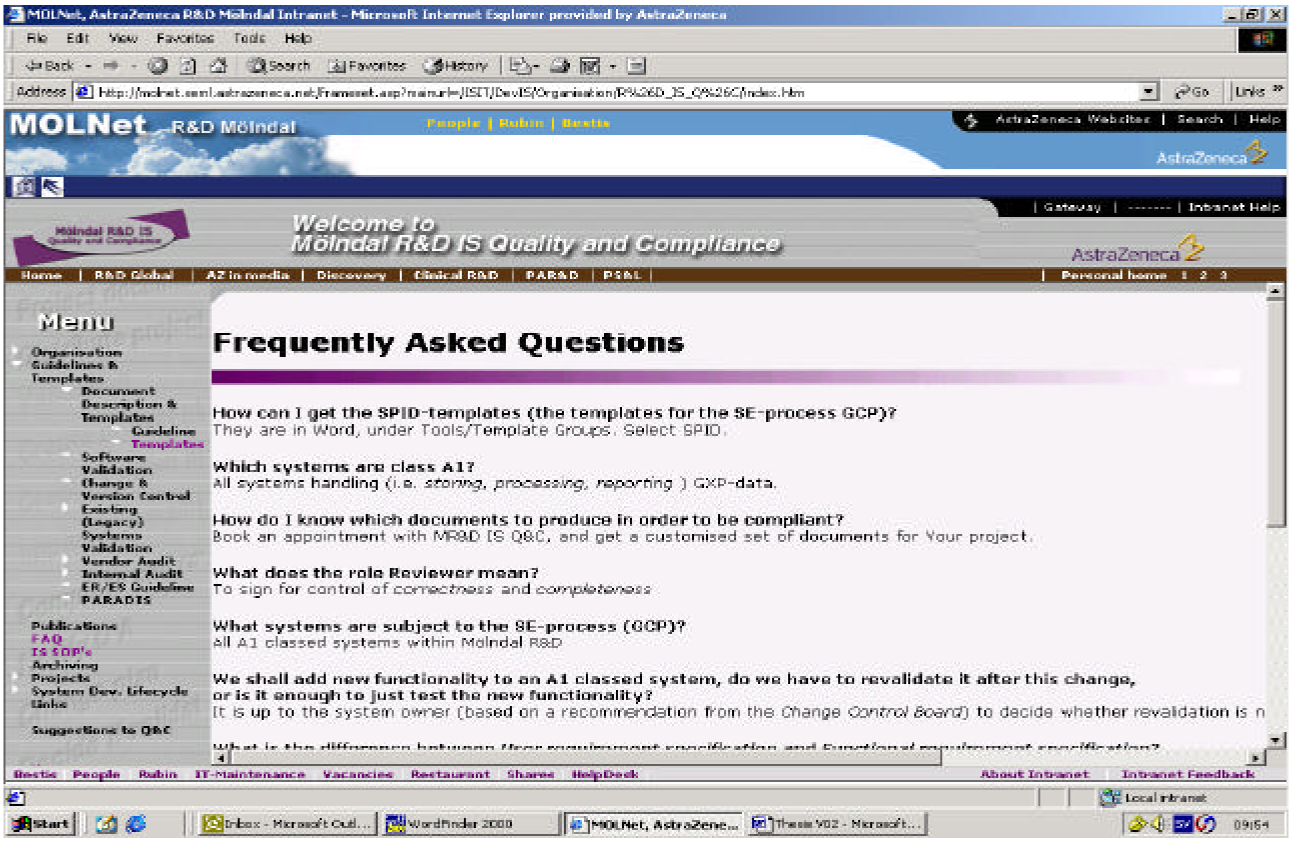

MOLNet, AstraZeneca R&D Mölndal Intranet - Microsoft Internet Explorer provided by AstraZeneca
Welcome to Mölndal R&D IS Quality and Compliance
Menu
Design
Back
Customisation meeting with MR&D IS Q&C
Project Specification
Distribute proj. spec. to B.QA
Project Plan
Quality Plan
Validation Plan
Distribute validation plan to B.QA
Test Plan
Training Plan
Validation Protocol 1
Functional Specification
Deviations?
No
Installation Requirement Specification
Validation Protocol 2
Deviations?
Go to sub-Process Implement

Figure III. The design process



Frequently Asked Questions

How can I get the SPID-templates (the templates for the SE-process GCP)?
They are in Word, under Tools/Template Groups. Select SPID.

Which systems are class A1?
All systems handling (i.e. *storing, processing, reporting*) GXP-data.

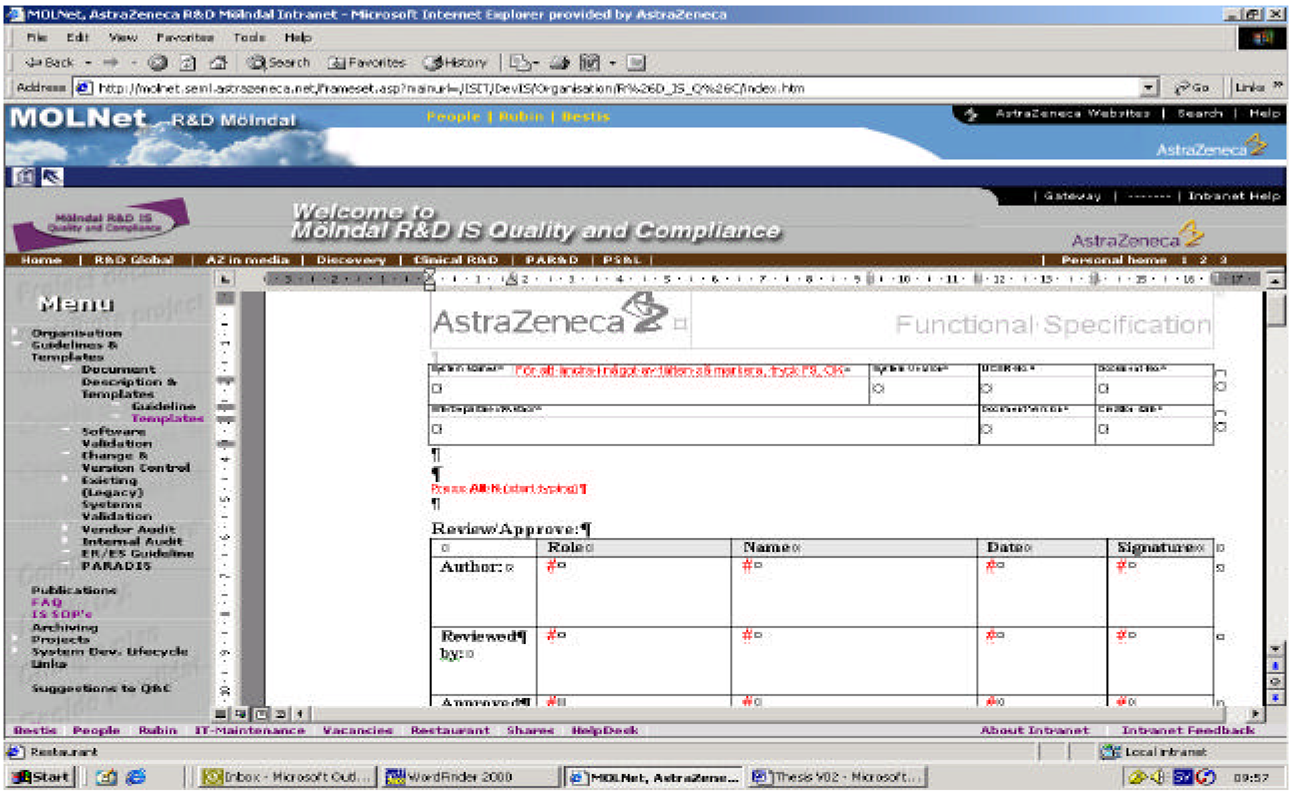
How do I know which documents to produce in order to be compliant?
Book an appointment with MR&D IS Q&C, and get a customised set of documents for Your project.

What does the role Reviewer mean?
To sign for control of *correctness* and *completeness*

What systems are subject to the SE-process (GCP)?
All A1 classed systems within Mölndal R&D

We shall add new functionality to an A1 classed system, do we have to revalidate it after this change, or is it enough to just test the new functionality?
It is up to the system owner (based on a recommendation from the Change *Control Board*) to decide whether revalidation is n

Figure IV. The FAQ



För att ändra i något av fälten så markera, tryck F9, OK

Press Alt-N (start typing)

Review/Approve:

	Role	Name	Date	Signature
Author:	#	#	#	#
Reviewed by:	#	#	#	#
Approved	#	#	#	#

Figure V. Documenting on-line through the EPL

The SPID Questionnaire
The questionnaire is divided into the following sections:

1. The training efforts (including 3 questions)
2. The SPI unit's feedback and support (including 5 questions)
3. The software validation guideline (including 4 questions)
4. The document description guideline (including 3 questions)
5. The templates (including 4 questions)
6. The change and version control guideline (including 7 questions)
7. The training efforts (including 3 questions)

The Networking Questionnaire
The following is a selection of the 19 questions included:

1. What is a network?
2. When shall we use networks?
3. Why shall we use networks?
4. What is the management role in networking?
5. How should networks be organised?

The Final Project Report for SPID 2001
The document consists of the following sections:

1 Introduction
1.1 Background and purpose
1.2 The products
1.3 Contents of document
2 The CMM assessments I & II
2.1 Assessment I
2.2 Assessment II
2.2.1 What should be improved?
2.3 The comparison between assessment I & II
3 Evaluation of implementation measures
4 Lessons learned and further improvement suggestions
5 Conclusion

Gothenburg Studies in Informatics, Report 22, March 2002, ISSN 1400-741X

Gothenburg Studies in Informatics
Department of Informatics
Göteborg University
Viktoriagatan 13, Box 620
SE – 405 30 Göteborg, Sweden
E-mail: reports@informatics.gu.se
WWW: www.informatics.gu.se