Scandinavian Journal of Information Systems

Volume 13 | Issue 1 Article 7

2001

ANALYZING an SPI project WITH THE MAP FRAMEWORK

Pouya Pourkomeylian
AstraZeneca R&D, pouya.pourkomeylian@astrazeneca.com

ANALYZING an SPI project WITH THE MAP FRAMEWORK

Pouya Pourkomeylian

AstraZeneca R&D Mölndal, S-431 83 Mölndal, Sweden Tel: +46 31 776 17 96, Fax +46 31 776 37 30 Viktoria Institute, Box 620, 495 30 Göteborg, Sweden Pouya.pourkomeylian@astrazeneca.com

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 provides 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. 2000). 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 Basili 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 2001).

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.

2.1. 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 the so-called GXP (Good Manufacturing Practice, Good Clinical Practice, and Good Laboratory Practice) rules. The GXP rules are the authorities' quality requirements for pharmaceutical companies to ensure patient health, the quality of production processes (e.g. clinical studies or software development activities) 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 the software development activities (e.g. software change control, software validation, and data processing and storage) are being performed in compliance with the quality requirements. Therefore every software project regulated by GXP requirements has to carefully apply all quality rules and has to be able to show by recorded 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 have to be applied for all information systems development and use regulated by the GXP requirements.

2.2. 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 & 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 2000), 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 2001). 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, omong 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 now being used in the first 12 software projects – which admittedly only represent a small amount of all projects running in

the company- and are appreciated by practitioners working in these software projects. The SPI unit measures and documents their 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.

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 Norbjerg (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.

ll 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 the SPID group 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 that 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 new processes in the organisation in the period June 2000-June 2001,

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.

Table 1 SPID activities based on the MAP

a qualitative feedback measurement mechanism was established based on the implementation plan of the project. This mechanism supports 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 2001).

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 overall development process including processes such as the software test process and the software configuration management process, which were 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 for practitioners' work was unknown. Now in March 2001 we are continuously getting 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 of not being able to maintain commitment to follow an evolutionary approach. However, as all group members were aware of the nature of an evolutionary approach and as everyone was interested in solving the software process problems, the group kept maintaining its commitments to the project after all. 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 strong motivation and enthusiasm among practitioners and management. 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 a complete view over all the 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 SPI group 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 field of software engineering developed during the course of the project. Although not all the practitioners could built up extensive 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 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 members and the reference group discussed different interpretations of a specific task also supported experience sharing and competence building. However, during the 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 and this could have had a negative impact on other practitioners and could have created resistance against using the new processes. As, however some effort has been put into providing information about the results of the project on a continuous basis throughout the project to keep other practitioners informed, this did not happen

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 the before mentioned 12 software projects. 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, a feedback measurement activity based on the results of practical use of the processes in these 12 software projects is now planned. Discussions with management concerning long-term improvements have also been initiated.

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., Ngwenyama O., (2001). A Conceptual MAP of Software Process Improvement, Scandinavian Journal of Information Systems (in this volume)

Andersson I., Nilsson K., (2000). Diagnosing Diffusion Practices within a Software Organisation, in Svensson, L. et .al (eds.), Proceedings of IRIS 23, pp. 393-406.

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

Arent J., Norbjerg J., (2000). SPI as Organizational Knowledge Creation: A Multiple Case Analysis. Proceedings Conference Proceedings at Maui Hawaii.

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

Goldenson, D. R., Herbsleb J., D., (1995). After the Appraisal: A Systematic Survey of Process Improvement, its benefits, and Factors that Influence Success. CMU/SEI-95-TR-009, SEI, Pittsburgh.

Herbsleb, J., et al., (1997). Software Quality and the Capability Maturity Model. Communications of the ACM, 40(6), 30-40.

Humphrey, W. S., (1998). Managing the Software Processes, Addison-Wesley Publishing Company, USA.

Johansen, J., Mathiassen L., (1998). Lessons learned in a National SPI Effort. EuroSPI 98 Göteborg.

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

Mashiko, Y., Basili 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, Software Engineering Institue, Cornegie Mellon University.

Paulk M C., et al., (1993). The Capability Maturity Model for software Version 1.1, Carnegie Mellon University, Software Engineering Institute.

Pourkomeylian P., (2000). Knowledge Creation in Improving a Software Organization, in Svensson, L. et .al (eds.), Proceedings of IRIS 23, pp.163-180.

Pourkomeylian P., (2001). An Approach to Institutionalisation of Software Processes, forthcoming in ISD 2001.

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

Tryde, S., Nielsen A.-D. & Pries-Heje J., (2000). A Framework for Organizational Implementation of SPI in Practice. In: L. Mathiassen et al. (Editors): Learning to Improve.

Zahran S., (1998). Software Process Improvement, Software Engineering Institute, SEI Series in Software Engineering, Addison Wesley Longman.