ORGANISATIONAL DYNAMICS IN THE SOFTWARE PROCESS IMPROVEMENT: THE AGILITY CHALLENGE

Anna Börjesson[1] and Lars Mathiassen[2]
[1]Ericsson and IT University of Gothenborg, Sweden; *[2]Georgia State University, USA*

Abstract: It is a challenge to keep up momentum in Software Process Improvement (SPI) during organizational changes. SPI initiatives get interrupted, are side-tracked and progress slowly when changes occur in the target organization. This paper explores this crucial relation between SPI initiatives and the dynamics of the organization to be changed. The study builds on longitudinal data from the introduction of a new approach to requirements management into a software unit within Ericsson. Our focus is on the challenges involved in managing the SPI initiative as a sequence of organizational changes occurs in software development. We discuss the findings from this study in relation to the SPI literature and the literature on organizational agility and software agility. Our research indicates that SPI can benefit from agility ideas if the innovations are integrated well with other agility initiatives within the software organization. We therefore suggest that there is a need to coordinate and align the software agility movement with SPI issues to arrive at a comprehensive and holistic understanding of how software organizations can respond effectively to dynamics in their environment.

Key words: Software Process Improvement, Agility, Organizational Dynamics, SPI Implementation Success

1. INTRODUCTION

Software Process Improvement (SPI) has been adopted by many organizations as a strategy to enhance their capability to deliver qualitative software (Humphrey, 1989; Grady, 1997; Mathiassen et. al, 2002). Although very successful cases of improvement have been reported (Humphrey et al., 1991; Haley, 1996; Diaz and Sligo, 1997; Larsen and Kautz, 1997) there is also a critical debate on this approach to improve software performance (Bach, 1995; Bollinger and McGowan, 1991; Fayad and Laitinen, 1997; Humphrey and Curtis, 1991). The most recent report from the Software Engineering Institute puts the rate of failure at around 70% (SEMA, 2002). A growing amount of research offers explanations and guidance to improve the success rate of SPI initiatives. McFeeley (1996) discusses the steps and activities an SPI initiative should include and presents the IDEAL model for SPI. Mashiko and Basili (1997) and Ravichandran and Rai (2000) examine the relationship between SPI and software quality management. Fichman and Kemerer (1997) discuss the organizational barriers towards adoption of software process innovations. Abrahamsson (2000, 2001) discusses the need to manage commitment from different stakeholders. Nielsen and Nørbjerg (2001) emphasize the need to focus on social and organizational issues in SPI. Aaen (2002) argues for helping software developers help themselves rather than having change agents drive SPI. Börjesson and Mathiassen (2003a) argue that SPI initiatives benefit from spending more of their effort in the later phases of the IDEAL model where focus is on deployment. While these contributions offer a broad range of perspectives on how to improve the SPI success rate and guide the SPI initiatives towards success, very little attention is paid to one of the key challenges faced by SPI initiatives: the dynamics of the software organization they target.

Software organizations constantly need to react to market dynamics, new customer requirements, technological innovations, and mergers between software companies. The degree and pace of the implied organizational dynamics have increased over the past years as indicated by the notions of fast-moving software organizations (Baskerville et al., 2001; Holmberg and Mathiassen, 2001) and radical IT-based innovations (Lyytinen and Rose, 2003). SPI initiatives are therefore increasingly faced with the challenge of developing and implementing improved processes into an organizational context that is constantly changing. To be successful they must be organized, managed, and executed in ways that take these organizational dynamics into account and address them effectively. The software engineering community has over the past years adopted agile philosophies and approaches to improve development practices in response to the increasing dynamics faced in software development (Abrahamsson et al., 2002). The theme of our research is to extend this notion of software agility by studying challenges related to improvement of software practices, i.e. SPI in dynamic organizational contexts.

The study is exploratory in nature. It combines an empirical study of SPI practices with insights from the literature on organizational agility and software agility. The purpose is to understand the challenges SPI initiatives face and the possible strategies and tactics they can adopt to deal with the dynamics the software organization is facing. We present data from a longitudinal case study during the period 2000 to 2003. The initiative was carried out in a product development unit within the Ericsson Telecom Company in Gothenburg, Sweden. The initiative focused on introducing a new requirements management approach and tool, Requisite Pro. We study the activities involved and the impact the new requirements management approach had on configuration control, baseline control, and traceability over time, while the software unit was subject to a number of organizational changes. Subsequently, we interpret the experiences from this initiative by relating them to the agility concept. The paper is structured around three main questions:

- What was the impact of organizational dynamics on the SPI initiative?
- In what ways could the initiative have benefited from adopting more agile SPI tactics?
- What are the implications for development of agile approaches within software engineering?

The argument is presented as follows. Section 2 presents the literature on SPI and organizational dynamics together with the literature on organizational agility and software agility. Section 3 describes the research approach. Section 4 presents data from the longitudinal case study with particular focus on the implications organizational dynamics had on the change initiative. Section 5 discusses the contributions of our research and implications for both practice and research. We conclude our findings in section 6.

2. THEORETICAL CONTEXT

Initially, we review the SPI literature with particular focus on how the issues related to organizational dynamics are addressed (Section 2.1). We then look closer at the literature on organizational agility that specifically aims to address challenges related to responding effectively to changes in the environment (Section 2.2). Finally, we look at how these ideas have been adopted to develop agile approaches to software development and open up for future research in the light of the organizational dynamics issues reported here (Section 2.3).

2.1 Organizational Dynamics in the SPI Literature

The relation between organizational dynamics and SPI success is not well investigated and understood in the core SPI literature. Parts of the SPI literature touch upon the challenges involved in dealing with organizational dynamics in SPI, but very little is said directly and the managerial tactics offered are few and vague. The Capability Maturity Model – CMM (Paulk et. al., 1995) is one of the most widely spread and used SPI models. CMM is structured in five levels of maturity and defined by a number of key process areas (KPAs). The idea is to work with the KPAs on the next level to address problems and improve practices in the organization. Very little is mentioned about organizational dynamics. The model assumes a relatively stable organizational environment while addressing KPAs to achieve higher CMM levels.

Humphrey (1989) discusses 'The Six Basic Principles' for SPI and one of them is 'Ultimately, everyone must be involved'. If improvement initiatives appear threatening to people, they will likely cause resistance to change. This implies that key stakeholders need to be involved to address the implications of organizational growth or the inclusion of new people into the organization. In the same way, McFeeley (1996) argues that management needs to be involved. He also specifically describes in the IDEAL model a step called 'Revise Organizational Approach'. The goal is to insure sponsorship for SPI from managers. This implies that SPI sponsorships need to be evaluated and negotiated when the software organization is re-organized. Grady (1997) emphasizes along the same lines the importance of 'Organizational Readiness', i.e. how much and how widespread the enthusiasm for change is. Weinberg (1997) argues in general that growth has a natural negative impact on quality. One of the reasons is that special initiatives are needed to ensure dedication to software development and commitment to change initiatives as new people become involved as a consequence of re-organizations.

Zmud (1982) discuss diffusion of new software practices. These practices, for instance configuration management and structured review, are typical examples of improvement areas that an SPI initiative drives. Zmud argues that the success depends on the organizational context and the resources in the organization whose behaviors need to change. Factors found to influence the diffusion of new software practices in one organizational context may be seen to have little or no impact in another organizational context. Fichman and Kemerer (1997) discuss these difficulties from an organizational learning perspective. Successful diffusion of software process innovations depends on the ability to facilitate organizational learning. This is especially relevant for complex organizational technologies such as object oriented programming languages.

More recent SPI literature (Holmberg and Mathiassen, 2001) discusses the challenges involved in combining a capacity for radical innovations while at the same time facilitating evolutionary improvements in software organizations. Holmberg and Mathiassen studied a fast-moving software organization and concluded a number of lessons learned to cope with a dynamic environment while simultaneously trying to improve professional practices. First, the authors argue that there is a reciprocal relationship between innovations and improvements that everyone in the organization must understand. Second, it is easy to ignore improvement work in favor of the more hype innovation work and the improvement culture must therefore be protected. Third, improvements must be conceived as relevant and useful in the organization. Finally, SPI practitioners must understand the core processes of the business they work in. Holmberg and Mathiassen argue that SPI is particularly important to integrate into fast-moving software organizations if they are to survive in the long run.

2.2 Organizational Agility

The agile movement is led by the Agility Forum at Lehigh University. The forum was formed in 1991 to implement the vision of a new system for production of goods and services through adoption and implementation of the agile organizational paradigm. Gunneson (1997) argues that even though there are different interpretations of organizational agility there are specific elements that are generally agreed upon. Agility is concerned with economies of scope, rather than economies of scale. Where lean operations are usually associated with efficient use of resources, agile operations are related to effectively responding to a changing environment while at the same time being productive. The idea is to serve ever-smaller niche markets and individual customers without the high cost traditionally associated with customization. The ability to respond is hence the essential and distinguishing feature of the agile organization (Dove, 2001). The reason that agility has emerged over the past decade as a key challenge for organizations is the increasing pace and unpredictability of changes in the economical, technological, and political environment (Gunneson, 1997; Dove, 2001).

We have chosen to focus on Dove's "Response Ability – The Language, Structure, and Culture of the Agile Enterprise". First, it emphasizes the complementary challenge of today's organization: "It must generate at least as much fuel as it consumes (profitability) and it must continuously adapt as necessary to changing environmental conditions (adaptability)" (Dove, 2001, p. 3). Second, it describes a set of elements that are required to successfully develop agile practices. Third, it offers explicit criteria by which one can evaluate the current organization as part of moving towards more agile practices.

Agility requires "the ability to manage and apply knowledge effectively, so that an organization has the potential to thrive in a continuously changing and unpredictable business environment" (Dove, 2001, p. 9). The two key elements that are required to practice agility are hence *response ability* and *knowledge management.* Response ability is achieved through change proficiency (process) and flexible relationships (structure) that are reusable, reconfigurable and scalable. Knowledge management in turn requires collaborative learning (process) and knowledge portfolio management (structure) including the identification, acquisition, diffusion, and renewal of the knowledge the organization requires strategically. Each of these elements are detailed and discussed in Dove's approach (2001) to the agile organization.

The core feature, response ability, is understood and assessed as illustrated in Figure 1. The opportunistic organization has high reactive proficiency, but it has weak proactive abilities. It follows best practices, listens to the customer, and is good at improving current capabilities. The innovative organization has high proactive proficiency, but it has insufficient reactive abilities. It can quickly introduce new technologies, services, strategies, and concepts to adapt to changing conditions in its environment. The organization is fragile when it has insufficient reactive ability to identify and explore opportunities related to its current capabilities as well as insufficient proactive ability to innovate as required by the environment. When market pressures are high and the environment is turbulent the ideal is the agile organization that combines high levels of reactive and proactive abilities.

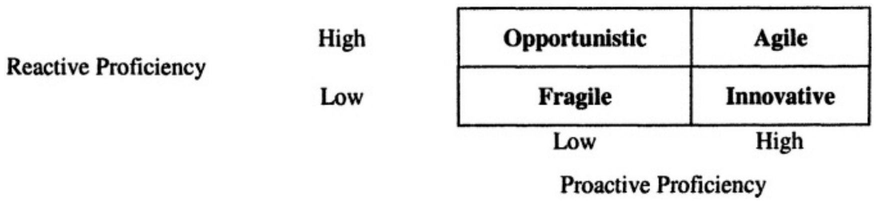

Figure 1. **Response Ability States (Dove, 2001)**

2.3 Software Agility

The agility concept has been adopted by the software engineering community and a recent review of the literature is provided by Abrahamsson et al. (2002). The introduction of extreme programming (Beck, 1999) is acknowledge as the key step in this development, but there are also other published methods like Crystal Methods (Cockburn, 2000), Feature-Driven Development (Palmer and Felsing, 2002), and Adaptive Software Development (Highsmith, 2000). The key idea is "the recognition of people as the primary drivers of project success, coupled with an intense focus on effectiveness and maneuverability. This yields a new combination of values and principles that defines an *agile* world view" (Highsmith and Cockburn 2001, p. 122). Several of the key actors involved have expressed these values in the Agile Software Development Manifesto (Beck et al. 2001):

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

The strategy adopted to practice these values is, according to Cockburn (2002), the use of light-but-sufficient rules of project behavior combined with the use of human- and communication-oriented rules. Agile processes are both light and sufficient. Lightness helps a project to remain maneuverable, responding effectively to changes. Sufficiency is a matter of staying in the game, responding effectively to customer needs. Cockburn more specifically proposes a number of tactics to implement such a strategy: two to eight people in each room to facilitate communication and community building; onsite user experts to establish short and continuous feedback cycles; short increments of one to three months to allow quick testing and repairing; fully automated regression tests to stabilize code and support continuous improvement; and experienced developers to speed up development time.

3. RESEARCH APPROACH

This research is part of a collaborative practice study (Mathiassen, 2002) carried out at one of Ericsson's system development centers with more than 20 years of experience developing packet data solutions for the international market. This particular Ericsson organization has grown from 150 employees in 1995 to 900 in 2001 and been re-organized and downsized during 2001 to 2003. SPI has during this dynamic period become an increasingly important area in order to ensure quality deliveries.

The two authors represent industry and academia in close cooperation to secure relevant data and an appropriate theoretical framing of the study. The overall purpose of the research collaboration was two fold. We wanted to improve SPI practices at Ericsson while at the same time contributing to the body of knowledge on SPI.

The paper addresses the implementation of a new requirement management approach during the turbulent period from 2000 to 2003. Though the studied SPI initiative is successful in the end it suffers from initial weak impact and too slow progress. One of the authors was working in and been responsible for the initiatives. The potential bias and subjectivity is handled both through the collaborative research methodology and through discussion and interviews with engineers and researchers both within Ericsson and from the research community. The research is based on a case study (Galliers, 1992; Yin, 1994) with a focus on process implementation and use to assure SPI success. The SPI unit collected the basic data during the initiative with help of the Requisite Pro tool, SPI reports, and SPI project specifications.

The identified problems were collected from SPI reports and discussions with practitioners who used the new way of working. These discussions were made possible through project participation by one of the authors. The evaluations are based on questionnaires with follow-up interviews and discussions with software line and project managers and carried out in collaboration with those involved in the initiatives. Eight line and project managers were asked to answer questions about the implementation and use of the Requisite Pro tool within different development projects (see Table 1). Follow-up interviews were made in five cases to clarify previous answers.

4. CASE

The studied case is about introduction of a new requirements management approach in one product development unit at Ericsson, Gothenburg, Sweden. The focus of the study is on the implementation and use of Requisite Pro and the related approach to requirements management. The use of this approach provides the organization with requirements configuration control, requirements baseline control, and traceability from requirements to test cases. The organization needed to improve requirements management to secure higher quality in the software parts of their products. There is a well-known potential in improving requirements management to secure better software quality (Humphrey, 1989).

4.1 The SPI Context

In late 1999 the SPI unit started to organize their work in dedicated initiatives supporting one software engineering project at a time (Börjesson and Mathiassen, 2003a). Projects that ran in partly overlapping sequence inherited the improved way of working from the previous project. In some cases the organization ran parallel software engineering projects that only partly differed in software and functionality. In these cases the SPI initiatives were coordinated through an SPI initiative that consisted of people from the two dedicated initiatives.

Each SPI initiative consisted of 7-8 part time process engineers that supported approximately 150 software engineers. Each initiative focused on several different software engineering practices such as requirements management, configuration management, project management and test management. 1-2 process engineers within each SPI initiative were dedicated to work with requirements management.

4.2 Adoption of Requirements Management

In late 1999, there was a management decision to adopt Rational Unified Process (RUP) (Krutchen, 2000) within the organization. The decision was a result of a major investigation comparing state-of-the-art software processes with respect to content, potential and coverage. One part of this decision aimed to change requirements management practices. Requisite Pro, a new way of describing requirements, use cases, and a new way of tracing requirements both to design and test was introduced. We have structured the introduction of the new requirements management approach into four phases (see Figure 2) corresponding to increased adoption of the approach. Each arrow represents a software engineering project that delivered one release of the product based on the approach. The phases are further explained in Table 1 (implementation success) and Table 2 (characteristics of the phases).

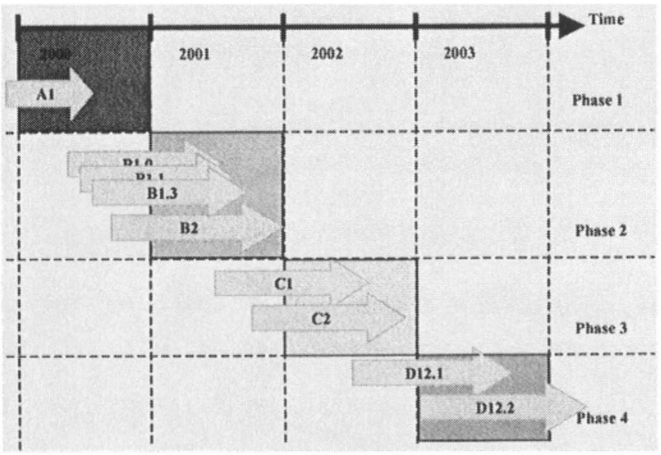

Figure 2. **Improvement phases for new requirements management approach**

Successful implementation of the requirements management approach is defined as the extent to which the tool was used to support control of requirements, i.e. baseline control, configuration control and test case traceability. Baseline control implies freezing a set of requirements artifacts as a baseline for all subsequent requirements activities. No changes can be made to those artifacts without a formal decision amongst the involved stakeholders. Configuration control means keeping track of versions of requirements artifacts, whether they are approved, and other status information. To have baseline and configuration control implies that all requirements artifacts are stored and visible at a place known to everyone needing them. Test case traceability means that all test cases can be traced to a specific requirement. Table 1 shows the implementation success of the requirements management approach in each phase with respect to baseline control, configuration control and test case traceability. We focus on the extent to which Requisite Pro supported requirements management. There were previously various manual controls of the requirements based on tools such as Excel.

Table 1. Implementation success of Requisite Pro (Project numbers refer to Figure 2)

Phase	Achieved SPI result	Configuration and Baseline control	Test case traceability
1	No focused and dedicated SPI initiative was organized. Several improvements in the requirements management area were made, e.g. adoption of spread sheets to control requirements manually.	A varying number of spread sheets stored within project areas or on local discs. No tool support to control requirements.	0%
2	All requirements in B2 were stored in the Requisite Pro tool, but only part of the test cases. B2 adopted tool support for configuration control of requirements.	Requisite Pro introduced and started to be used in B2. The time pressure in the B1.x releases made the introduction slower. Project management did not use the tool for follow-up.	20% (based on interviews with responsible managers)
3	The requirements management approach was improved based on experiences from B2, The C2 project managed to use the tool to support most test cases.	Requisite Pro further diffused and used. More features were adopted from Requisite Pro to provide more information. Many engineers did still not understand why they should provide information to Req Pro.	50% (based on interviews with responsible managers)
4	D12.1 managed to achieve baseline and configuration control as well as full support of test case traceability. D12.2 managed to get full control from the start of the project.	More or less 100% use of Requisite Pro by all engineers. The tool used by management for active follow-up.	99,7% (based on measures in Req Pro)

4.3 The Change Process

Each software engineering project described in Figure 2 and Table 1 benefited from the SPI initiative, but the speed and effectiveness of the change process is questionable. Why were the software projects in phase 2 or 3 not capable of achieving the level of tool-support that was reached in phase 4? To understand in more detail how the SPI initiatives worked during the different phases, Table 2 shows data for each phase: what the main problems were, how the SPI initiative was organized, which practitioners that were involved, and how results were achieved.

Table 2. Characteristics of the SPI initiative

Phase	Problem	SPI initiative	Involvement	Result
1	• Requirements documents stored on local discs • Adoption of ad-hoc spread sheets • Low tool support to cover test	• No formally organized initiatives. Several attempts were made by the software project to get all spread sheets under control.	• Requirements engineers • Test engineers	• Poor tool support for baseline control • Poor tool support for configuration control • Poor tool support for traceability
2	• Low Requisite Pro competence in software projects • Few competent users made administration of Requisite Pro hard • Major resistance to change • Low involvement from requirements engineers • Technical difficulties with Requisite Pro	• Process engineers and external consultants defined the requirements strategy, held courses, and supported the software projects.	• Requirements engineers • Test engineers • Process engineers • External consultants	• Requisite Pro was introduced and used by a few dedicated individuals. Project B2 achieved tool support for configuration control of requirements. Still no baseline control. • •Test case traceability was improved, but still low.
3	• New staff involved • New managers • Major resistance among the new people	• Senior engineers were involved to a higher degree • Management became involved and took responsibility for tool adoption	• Requirements engineers • Test engineers • Process engineers • Managers	• Tool supported configuration control • Tool partially supported base line control • Improved support of test case traceability
4	• New SPI staff involved • New technical difficulties • New category of engineers involved	• High involvement from engineers • The tool was introduced as a management tool to support control of requirements	• Requirements engineers • Test engineers • Process engineers • Managers • Design engineers	• Tool supported configuration control • Tool supported base line control • Tool supported test case traceability

No major improvements were implemented based on the new requirements management approach in the first phase. All focus was directed towards getting manual configuration control of requirements and test cases. The implementation success of the new requirements management approach improved in a stepwise fashion in phases 2, 3 and 4. Figure 3 illustrates this change process graphically related to the four phases. The general expectation was that the organization should benefit from the implementation more progressively when the requirements management initiative was launched. This is illustrated by the dotted line. The question is why this didn't happen and what barriers could be reduced and enablers improved to speed up the change process. What were the main reasons for not reaching higher success in phase 2 or 3?

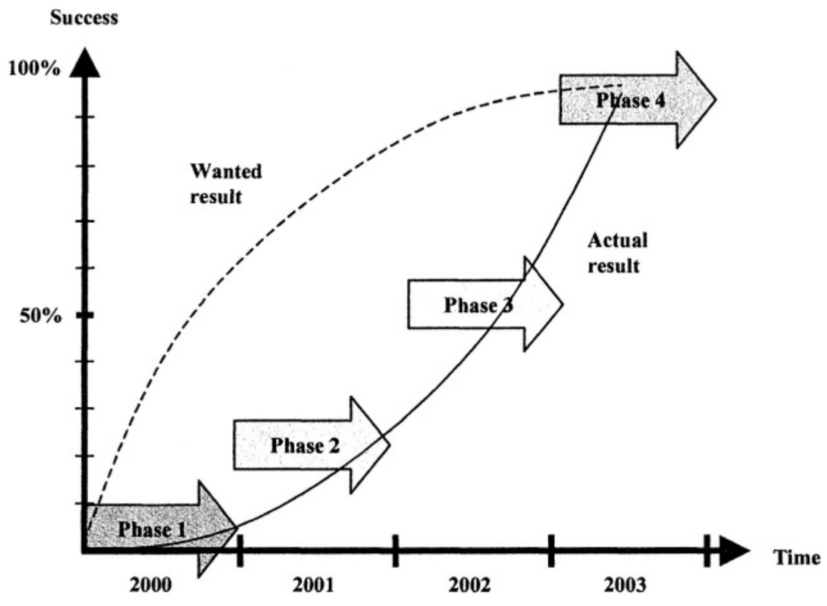

Figure 3. The adoption of the new requirements management approach

4.4 Organizational Dynamics

SPI initiatives are exposed to various forms of change reactions (Humphrey, 1989; McFeeley, 1996; Fichman and Kemerer, 1997; Grady, 1997). Weinberg suggests that SPI initiatives should be seen as sequences of reactions to attempts to change engineering practices from old status quo to new status quo (1997). The change process will go through a chaotic period and different attempts to integrate the new process into current practices.

During these events practices might return to old status quo or they might reach a sustainable, new status quo. This understanding of the change process is illustrated in Figure 4 and adopted to our case in Figure 5. Figure 4 also shows a simplified change curve to visualize the possible change reaction in Figure 4 and 5.

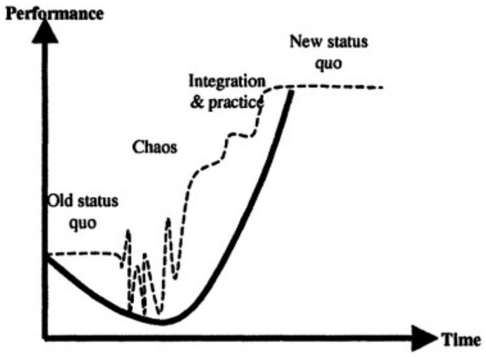

***Figure 4.* The simplified change curve**

In each phase, Weinberg's reaction to change occurred. The underlying reasons are partially revealed in Table 1 and Table 2: degree of practitioner involvement, tool difficulties, and degree of management commitment. These are all reasons acknowledged in the SPI literature (Humphrey, 1989; Grady, 1997; Abrahamsson, 2000). These characteristics, however, only partially explain why the change process progressed slowly and with little effect. Figure 5 includes an additional explanation: the organizational dynamics of the target software organization. The Y-axis 'Success' in Table 5 and 'Performance' in Table 4 both indicate the same thing, i.e. getting positive effects from the change.

Two major re-organizations occurred during the change process and interfered with the improvement initiative as it was progressing well. In late 2001, two local development companies were merged together to save money by sharing administrative units and concentrating development on future strategic products for Ericsson. In late 2003, two divisions within the development company were merged to achieve higher integration between different parts of the product by working closer together and using the same processes. The idea was also to reduce overhead from line and project management. Of the 200 new employees that were added to the software organization between phase 2 and 3, approximately 50 were affected by the new requirements management approach, and of the 150 new employees that were added between phase 3 and 4, approximately 100 were affected. Many integration activities took place such as general training in the product and the RUP processes, the development tools used, and the market that was targeted by the developed software. These integration activities were in the first merger (late 2001) managed by a special initiative that planned all general training. In the second merger (late 2002), an initiative was started to align the software processes used by the two organizations, which resulted in a fairly well described process. This process was, however, not sufficiently deployed. Furthermore, no attention was directed towards the ongoing SPI initiatives that at those points in time still weren't fully accepted and seen as a natural part of the daily practices in the software projects.

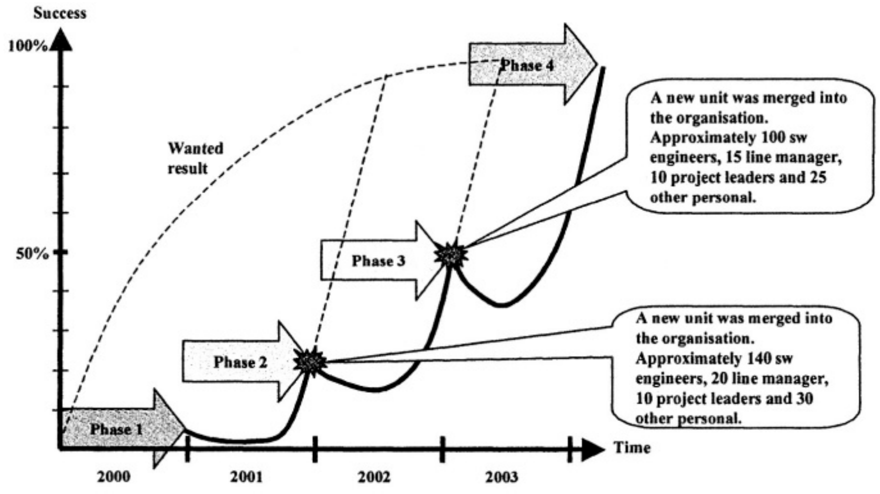

Figure 5. The interrupted change curve

5. DISCUSSION

In the following, we discuss and analyze the case and relate it to relevant literature on SPI and agility. The section is structured as a response to our research questions followed by a discussion of the limitations of the findings.

What was the impact of organizational dynamics on the SPI initiative?

SPI research has identified a number of relevant issues that correlate with the findings from this case study. Humphrey (1989), McFeeley (1996), Grady (1997) and Abrahamsson (2000) all argue for the need of management commitment to assure successful SPI. Table 1 reveals that management was insufficiently involved to facilitate successful implementation of the requirements management approach at an earlier point in time than phase 4. Moreover, senior engineers were not involved before phase 3 and the SPI work was driven by process engineers and process experts in the first phase. Involvement of senior engineers and close collaboration between software and process engineers is important to create a good mix of practice pull and process push (Börjesson and Mathiassen, 2003b). Iteration has also been recognized as a useful approach to SPI where learning from previous iterations can leverage successful implementation in later iterations (McFeeley, 1996; Grady, 1997; Börjesson, 2003). One interpretation of the case is that successful implementation of the new requirements approach simply required a number of iterations involving different software engineering projects. As the SPI work iterated, lessons were learned, more features of the tools and templates were used, more roles became involved, and the initial requirements approach was as a consequence improved. Interpretations like these are well in line with the literature on SPI and organizational change and the case can in this way be said to confirm existing knowledge.

The most interesting aspect of this case is, however, that two major organizational changes occurred in the software unit at Ericsson during the SPI initiative, see Figure 5. Between phase 2 and 3 approximately 50 new employees and between phase 3 and 4 approximately 100 new employees were affected by the requirements management approach. From the data in Table 2, we see that no major attention was paid towards integrating these individuals and the different engineering traditions that they represented into the ongoing improvement work. The SPI initiative itself demonstrated no ability to sense and respond to these organizational changes, only organization-level responses were launched. During the first merger there was general training in the product and RUP, in the development tools, and in the market that was targeted by the developed software. During the second merger, a dedicated initiative was launched to align the different software processes used by the two organizations. The latter initiative resulted in a fairly well described process; but the process was never fully deployed.

The two mergers and the related organizational changes had serious implications for the SPI initiative because new engineers with quite different backgrounds had to accept, learn and adopt the new requirements approach. We know that organizational dynamics affect personal behaviors and the ability to execute improvement initiatives successfully (Zmud, 1982; Weinberg, 1997; Holmberg and Mathiassen, 2001). The new employees that were added as a result of the mergers were, however, not integrated into the SPI work so they could influence and contribute to the improvement work and the quality of the results. They therefore reacted passively or negatively to the SPI initiative (Weinberg, 1997; McFeeley, 1996). At the same time, no response was launched from the SPI initiative targeting this challenge. This task was left to general initiatives that were launched independent of the SPI project. The independent initiatives did, however, not effectively resolve the issues implied by the two mergers. The ongoing SPI initiative was as a result not capable of addressing the new employees and the organization as a whole was not capable of managing organizational mergers and improvement initiatives in parallel (Holmberg and Mathiassen, 2001). The organizational dynamics of the targeted software units had for these reasons quite a negative impact on the SPI initiative resulting in loss of momentum and a very slow and ineffective implementation process.

In terms of response ability, see Figure 1, the proactive proficiency demonstrated by the SPI organization was in this case low. There were no sense and respond mechanisms in place to help the SPI initiative identify, assess, and cope with the two mergers. The reactive proficiency of the SPI organization was at most medium. Activities did take place in response to the two mergers, but they were not integrated with the SPI initiative and they had little positive effect. These interpretations point in direction of a fragile or at best opportunistic SPI response ability in the presented case.

In what ways could the initiative have benefited from adopting more agile SPI tactics?

Dove (2001) defines the two key elements required to practice agility as response ability and knowledge management. Response ability requires change proficiency and flexible relationships and knowledge management is facilitated by knowledge portfolio management and collaborative learning (Dove, 2001). Table 3 offers an analysis of how the Ericsson SPI initiative could have benefited from more agile tactics along each of these dimensions. This analysis shows how the analyzed SPI initiative, viewed in isolation, could have benefited from higher proactive proficiency, greater flexibility towards inclusion of new employees, better and more explicit presentation of the RUP knowledge management facilities, and increased collaborative learning. While such tactics would likely have helped *the SPI initiative* to cope more effectively with the two mergers, the question remains of how to better align and coordinate across different activities within *the organization as a whole* to increase its response ability to changes in the environment.

Table 3. Analysis of organizational agility dimensions

Response Ability	Change Proficiency (the ability to proactively and reactively respond)	The SPI organization had weak to medium reactive proficiency (capability to react to the two mergers), but weak proactive proficiency (capability to sense, assess, and create responses to changes). Dove calls this state fragile or opportunistic. Such organizations do not innovate processes and tools as a proactive response to emerging changes and future needs. The lack of being able to identify such dynamics gives the SPI organization limited ability to respond effectively to change. *The SPI organization would have benefited from proactively including issues related to the mergers into each of its ongoing improvement initiatives. The SPI initiative would have benefited from explicitly analyzing and responding to the risks and issues that resulted from the two mergers.*
	Flexible Relationships (scalable and reusable relationships)	The process infrastructure and relationships between improvement initiatives did easily accommodate new people into the organization and thereby facilitate their learning and gaining from their experience and knowledge. There were, however, no sense mechanisms in place to make the SPI initiative aware of the emerging problems. Moreover, the SPI initiative had no readiness to involve additional competencies and assure commitment from new colleagues. *A proactive and more open attitude towards new colleagues and an ability to identify them as valuable resources rather than as problems would have benefited the ongoing SPI initiative.*
Knowledge Management	Knowledge Portfolio Management (the organization's management of core competencies)	Ericsson had an up-to-date infrastructure for managing knowledge. Their adaptation of RUP was available, supported and managed. However, the strategy for how to benefit from the RUP adaptation was not integrated into the SPI initiative to support the new employees, who came from other cultures with other methods and tools. *The organization would have benefited from integrating the RUP strategy and the related knowledge management practice better into the SPI initiative to help new employees understand and make use of the RUP adaptation.*
	Collaborative Learning (the support for collaborative learning net-works and events)	Fichman and Kemerer (1997) argue that successful diffusion of software process innovations depends critically on the ability to facilitate organizational learning. The general and independent knowledge management activities that took place in the software units had little or no effect on the SPI initiative. More importantly, no systematic attempt was made to link the knowledge already embedded into the ongoing SPI initiative with the knowledge of the new software engineers. *A focused and strong involvement of the new engineers into the ongoing implementation of the new requirements management approach would have resulted in additional learning and commitment to the new practices from the new software engineers.*

What are the implications for development of agile approaches within software engineering?

The agility concept has so far been adopted by the software community as summarized in (Abrahamsson et al., 2002) and expressed in the Agile Software Development Manifesto (Beck et al, 2001). One of the values, to respond to change over following a plan, is an expression of the response ability of agile organizations (Dove, 2001). This research suggests, however, that the current adoption of the agility concepts within software engineering is restricted in a couple of ways. First, the focus is mainly on responses to changes in customer demands. Other forms of dynamics realted to technological innovations, market changes, and organizational mergers are not explicitly addressed. The presented case shows, however, a need to adress software practices, SPI, and organizational mergers in an integrated fashion. Second, the ability to respond is tied directly to the operational level of developing software, i.e. to the software project level. There are few attempts to link software project agility to issues related to SPI or to other forms of organizational dynamics than changes in customer demands.

Similarly, there is within the SPI literature plenty of focus on issues related to change. Humphrey (1989) discusses the need of change agents and change management skills to successfully execute SPI and Weinberg (1997) has introduced the change model illustrated in Figure 4 and emphasizes the need for change artistry in SPI. There is, however, as we have argued no explicit attempts within SPI to address issues related to organizational dynamics in SPI. The presented case from Ericsson demonstrates the impact that organizational changes can have on SPI innitiatives and the possible advantages of adopting agile tactics within SPI to cope effectively with changes without severe loss of momentum and progress.

Our research suggests, however, also that organizational dynamics can seldom be dealt with effectively in isolation. Organizational disruptions typically penetrate many activities within the organization and they call for coordination and alignment of responses across functions and organizational levels. We suggest therefore, that agile software development and agile SPI are important contributions to dealing with the dynamics that software organizations face; but the overarching challenge is for the software community to rethink and develop its approach to software agility. The present focus on agile software development needs to be developed and integrated with the challenges related to agile SPI and both needs to be conceived as elements in agile software organizations more generally, see Table 4. We suggest - in line with the organizational agility idea (Gunneson, 1997; Dove, 2001) - that such a holistic and integrated software agility concept is needed if software organisations are to cope effectively with the increased speed of innovation, the need to continously improve current practices, and, last but not least, to respond effectively to emerging customer demands through delivery of quality software. More research is needed to develop this integration between streams of thinking and practice and to arrive at a holistic and comprehensive understanding of the agile software organization.

Table 4. The agility challenge

Organizational focus	Agility Challenge
Software Development	• To respond to changing customer demands • To participate in and adopt software process improvements • To sense and respond to technological innovations and market dynamics
Process Improvement	• To sense and respond to changes in software development • To sense and coordinate with other change initiatives • To provide a flexible process infrastructure
Software Organization	• To balance and coordinate development, improvements and innovations • To develop appropriate infrastructures and knowledge management practices • To develop response ability, organizational learning and metrics and in support of agile practices

The presented study has limitations. Most importantly, the case has focused on the relationship between SPI and organizational dynamics without including other dynamics between software organizations and their environment. There are also other factors, like culture and leadership, influencing any change effort than the ones reported here. Furthermore, case study research always implies biases from the specific environment in which it is conducted. It is therefore important to stress that the results from this research provide suggestions and indications rather than firm conclusions. Future research into organizational agility within the software industry will hopefully increase our understanding for how software projects, SPI initiatives, and software organizations at large can benefit from more agile strategies and tactics.

6. CONCLUSIONS

This study has analyzed longitudinal data from the introduction of a new approach to requirements management into a software unit within Ericsson during 2000-2003. Our focus has been on the challenges involved in managing the SPI initiative as organizational changes occurred in the unit.

The research has made an effort to address the questions: What was the impact of organizational dynamics on the SPI initiative? In what ways could the initiative have benefited from adopting more agile SPI tactics? What are the implications for development of agile approaches within software engineering?

A number of already identified findings such as the necessity of commitment to SPI, the involvement of senior engineers, and the benefits of iterative approaches are further supported by this study. New insights into the relationship between SPI and organizational dynamics suggest, however, that limited agile capabilities have a severe and negative impact on the momentum and progress of SPI initiatives. The SPI initiative in question was not able to effectively respond to the need for integrating new employees. The SPI organization's response ability was fragile or at best opportunistic with low to medium reactive proficiency, and low proactive proficiency.

It is likely that the organization would have been capable of reaching SPI success faster if it had used more agile SPI tactics, such as addressing the organizational changes directly as part of the SPI initiative by involving and educating the new employees in the ongoing SPI work. Our study suggests, however, that it is important to approach organizational dynamics issues in a coordinated and coherent fashion. Further research is therefore needed to include SPI issues and other challenges related to managing organizational dynamics into the software agility movement. The software industry is advised to rethink and more actively adopt agile strategies and tactics to respond effectively to changes in their environments and to manage ongoing SPI work in parallel with other improvement and innovation initiatives.

REFERENCES

Aaen I. (2002) Challenging Software Process Improvement by Design. Proceedings of ECIS 2002 The European Conference on Information Systems, Gdansk, Poland, June 6-8.

Aaen, I., Arent, J., Mathiassen, L. and Ngwenyama, O. (2001) A Conceptual MAP of Software process Improvement. Scandinavian Journal of Information Systems, Vol. 13, 123-146.

Abrahamsson, P. (2000) Is Management Commitment a Necessity After All. In: Software Process Improvement? Euromicro '00, Maastricht, The Netherlands, IEEE Computer Society, 246-253.

Abrahamsson, P. (2001) Rethinking the Concept of Commitment in Software Process Improvement. Scandinavian Journal of Information Systems Vol. 13, 69-98.

Abrahamsson, P., Salo, O. and Ronkainen, J. (2002) Agile Software Development Methods – Review and Analysis, Oulu, Finland: VTT Electronics, Publication #478.

Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B. and Slaughter, S. (2001). How Internet Software Companies Negotiate Quality. IEEE Computer, Vol. 14, No. 5, 51-57.

Bach, J. (1995) Enough About Process: What We Need are Heroes. IEEE Software, Vol. 12, No. 2, 96-98.

Beck, K. (1999) Extreme Programming Explained: Embracing Change. Reading, Massachusetts: Addison-Wesley.

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

Bollinger, T.B and McGowan, C. (1991) A Critical Look at Software Capability Evaluations. IEEE Software, Vol. 8, No. 4, 25-41.

Börjesson, A. (2003) Making SPI Happen: Iterate Towards Implementation Success. European Software Process Improvement Conference 2003.

Börjesson, A. and Mathiassen, L. (2003a) Making SPI Happen: The IDEAL Distribution of Effort. Hawaiian International Conference on System Science.

Börjesson, A. and Mathiassen, L. (2003b) Making SPI Happen: The Road to Process Implementation. Proceedings of the Twenty-sixth Information Systems Research Seminar in Scandinavia.

Cockburn, A. (2000) Writing Effective Use Cases. The Crystal Collection for Software Professionals. Reading, Massachusetts: Addison-Wesley.

Cockburn, A. (2002) Agile Software Development. Boston, Massachusetts: Addison-Wesley.

Diaz, M. and Sligo, J. (1997) How Software Process Improvement Helped Motorola. IEEE Software, Vol. 14, No. 5, 75-81.

Dove, R. (2001) Response Ability – The Language, Structure, and Culture of the Agile Enterprise. New York, New York: Wiley.

Fayad, M. E. and Laitinen, M. (1997) Process Assessment Considered Wasteful. Communications of the ACM, Vol. 40, No. 11, 125-128.

Fichman, R. G. and Kemerer, C. F. (1997) The Assimilation of Software Process Innovations: An Organizational Learning Perspective. Management Science, Vol. 43, No. 10, 1345-1363.

Galliers, R. D. (1992) Choosing Information Systems Research Approach, R.D. Galliers, 1992, in: R. Galliers, editor. Information Systems Research: Issues, Methods and Practical Guidelines, Blackwell Scientific Publications, Oxford.

Grady, R. B. (1997): Successful Software Process Improvement. Upper Saddle River, New Jersey: Prentice Hall.

Gunneson, A. O. (1997) Transitioning to Agility – Creating The 21st Century Enterprise. Reading, Massachusetts: Addison-Wesley.

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

Highsmith, J. (2000) Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. New York, New York: Dorset House Publishing.

Highsmith, J. and Cockburn, A. (2001) Agile Software Development. The Business of Innovation. IEEE Computer, Vol. 34, No. 9, 120-22.

Holmberg, L. and Mathiassen, L. (2001) *Survival Patterns in Fast-Moving Software Organizations.* IEEE Software, Vol. 18, No. 6, 51-55.

Humphrey, W. S. (1989) Managing the Software Process. Reading, Massachusetts: Addison Wesley.

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

Humphrey, W. S. and Curtis, B. (1991) Comments on "A Critical Look at Software Capability Evaluations". IEEE Software, Vol. 8, No. 4, 42-46.

Krutchen, P. (2000) Rational Unified Process An Introduction Second Edition. Addison-Wesley Professional.

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

Lyytinen, K. and Rose, G. M. (2003) The Disruptive Nature of Information Technology Innovations: The Case of Internet Computing in Systems Development Organizations. MISQ, Vol. 27, No. 4, 557-595.

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

Mathiassen, L. (2002) Collaborative Practice Research. Information, Technology & People, Vol. 15, No. 4, 321-345.

Mathiassen, L., Nielsen, P. A. and Pries-Heje, J. (2002) Learning SPI in Practice. In: Mathiassen et al. (Eds.) Improving Software Organizations. From Principles to Practice. Upper Saddle River, New Jersey: Addison-Wesley.

McFeeley, B. (1996) IDEAL: A User's Guide for Software Process Improvement. Pittsburgh: SEI. Handbook, CMU/SEI-96-HB-001.

Nielsen, P. A. and Nørbjerg, J. (2001) Software process maturity and organizational politics. In Russo, N. L., Fitzgerald, B., and J. I. DeGross (Eds.) Realigning Research and Practice in Information Systems Development: The Social and Organizational Perspective. Boston. Massachusetts: Kluwer.

Palmer, S. R. and Felsing, J. M. (2002) A Practical Guide to Feature-Driven Development. Upper Saddle River, New Jersey: Prentice-Hall.

Paulk, M. C., C. V. Weber, B. Curtis, and M. B. Crissis (1995) The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, Massachusetts: Addison-Wesley.

Ravichandran, T. and Rai, A. (2000) Quality Management in Systems Development: An Organizational System Perspective, MISQ, Vol. 24, No. 3, 381-415.

Weinberg, Gerald M. (1997) Quality Software Management Volume IV – Anticipating Change. New York, New York: Dorset House Publishing.

Yin, R. (1994) Case Study Research. Newburry Park, California: Sage Publications.

Zmud, R.W (1982). Diffusion of Modern Software Practices: Influence of Centralization and Formalization. Management Science, Vol. 28, No. 12, 1421-1431.