4 NAVIGATING SOFTWARE PROCESS IMPROVEMENT PROJECTS

Ivan Aaen
Aalborg University
Aalborg, Denmark

Anna Börjesson
IT University of Gotenburg
Gotenburg, Sweden

Lars Mathiassen
Georgia State University
Atlanta, GA U.S.A.

Abstract *Software process improvement (SPI) is one of the most widely used approaches to innovate software organizations. In this study, we identify and compare two different tactics for SPI projects. The first tactic, the supertanker, is inspired by centralist thinking. It is driven by process-push, and it aims for efficient process development and diffusion. The second tactic, the motorboat, is inspired by decentralist thinking. It facilitates practice-pull, and it aims for adaptive process development and diffusion. Our analysis of 18 SPI projects at Ericsson in Gothenburg shows how the two tactics lead to different practices and outcomes. We discuss on that basis what SPI tactics to use and identify the presence of muddy and unknown waters as the key characteristic that requires motorboat tactics. We suggest that today's changing business environment calls for agile SPI practices that employ adaptive governance mechanisms at the corporate level and combines motorboat tactics with revised supertanker tactics.*

Keywords Software process improvement, process implementation, project tactics, agility

1 INTRODUCTION

Software process improvement (SPI) is widely acknowledged as a viable strategy to enhance organizational capability to deliver qualitative software (Grady 1997;

Humphrey 1989; Mathiassen et al. 2002). Many SPI initiatives struggle, however, with low SPI success rates (Aaen 2002; Börjesson and Mathiassen 2004c; Fayad and Laitinen 1997), and several approaches to increase the SPI success rate have been suggested including the use of models for SPI (Paulk et al. 1995), the use of models for process diffusion (Pries-Heje and Tryde 2001), emphasis on SPI change agent capabilities (Humphrey 1989; McFeeley 1996), and management of reactions to change (Weinberg 1997).

The theme of this paper is agility (Dove 2001; Haeckel 1995, 1999). During the last decade, the speed of change in the software industry has increased (Baskerville et al. 2001; Holmberg and Mathiassen 2001). To be successful, software organizations must increasingly be organized, managed, and executed in ways that allow them to effectively sense and respond to unpredictable events in their environment. While the software discipline has focused extensively on software development agility (Abrahamson et al. 2002), there has so far been little focus on software organization agility and agility in SPI (Börjesson and Mathiassen 2004b). To learn more about SPI agility, we identify and compare two different SPI project tactics and study their use and outcome in 18 SPI projects within the telecom company Ericsson AB in Gothenburg, Sweden. One tactic is generic, aiming for the development of organization-level software process elements. We call this the *supertanker* tactic. The other tactic is dedicated, aiming for unique solutions for particular development projects. We call this the *motorboat* tactic. Based on this distinction we focus on the following research questions: *How do different SPI tactics affect the outcome and the ability to navigate in an ever-changing business environment*?

We start by presenting our theoretical framework and research method. Following this, we present and analyze the 18 SPI initiatives at Ericsson. The analysis shows how the two tactics led to different practices and outcomes. We use these experiences to discuss choice of tactics and development of agile SPI practices in software organizations.

2 THEORETICAL BACKGROUND

We first explicate key ideas that underpin the capability maturity model (CMM). We then review the agility mindset and its adoption in software organizations and SPI. Finally, we explicate the two SPI tactics that drive our research into practices at Ericsson.

2.1 The CMM Mindset

The CMM mindset is rooted in the CMM (Paulk, Curtis et al. 1993; Paulk et al. 1995; Paulk, Weber et al. 1993) and the current CMMI–CMM Integration (CMMI Product Team 2002a, 2002b). These models assume that good processes are predictable and will lead to good products.

To Humphrey (1989) a software process is the set of tools, methods, and practices used to produce a software product. Process management strives to produce software

according to plan while at the same time improving capabilities. Referring to TQM (total quality management) author Deming, Humphrey identifies statistical process control principles as basic to SPI:

> When a process is under statistical control, repeating the work roughly the same way will produce roughly the same result. To obtain consistently better results, it is thus necessary to improve the process. If the process is not under statistical control, sustained progress is not possible until it is (p. 3).

The focus is on function—what developers should do. The wider context with issues like customer, product, organization, leadership, and commitment plays a dominant role in TQM approaches (Creech 1994). While these issues are discussed in the SPI literature, they play minor roles in the CMM. SPI projects are predominantly viewed as internal to the software organization. In that sense, SPI efforts are done for, with, and by the software developing organization.

Today, the CMMI still portrays process as key to improvement and adopts process modelling as a dominant theme in SPI. Humphrey describes a procedure in a software process as a "defined way to do something, generally embodied in a procedures manual" and he likens this definition with the procedure concept used in programming (p. 158).

This line of thought traces back to Osterweil's (1987) highly influential paper entitled "Software Processes Are Software Too," published around the time Humphrey developed the first version of CMM. The earliest impetus for process programming arose from Osterweil's meetings with Humphrey and his team at IBM in the early 1980s (Osterweil 1997). Osterweil (1987) advocated

> that we describe software processes by "programming" them much as we "program" computer applications. We refer to the activity of expressing software process descriptions with the aid of programming techniques as process programming, and suggest that this activity ought to be at the center of what software engineering is all about (p. 4).

Feiler and Humphrey (1991) adopt a similar view, arguing that processes have many artifacts in common with software and require similar disciplines and methods. They suggest thinking about SPI in software development terms.

These ideas persist in today's CMMI.

- A *process description* is defined as a

  > documented expression of a set of activities performed to achieve a given purpose that provides an operational definition of the major components of a process. The documentation specifies, in a complete, precise, and verifiable manner, the requirements, design, behavior, or other characteristics of a process. It also may include procedures for determining whether these provisions have been satisfied (CMMI Product Team 2002a, p. 556).

- A *process element* is "the fundamental unit of a process. A process may be defined in terms of subprocesses or process elements. A subprocess can be further decomposed; a process element cannot" (CMMI Product Team YEAR, PG).

The CMM mindset is rooted in process thinking and modeling with an ambition to create sustainable progress by bringing processes under statistical control. The functional perspective on software practices focuses on internal aspects of software practices and the emphasis on control outweighs a concern for ability to respond to change.

2.2 The Agility Mindset

The increased speed of technology and market change has led to considerable interest in how organizations can manage and respond to unpredictable changes in their environment (e.g., Dove 2001; Gunneson 1997; Haeckel 1995, 1999). The Agility Forum at Lehigh University was formed in 1991 to develop new approaches for production of goods and services based on agile practices. Gunneson (1997) argues that 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 the essential and distinguishing feature of agile organizations (Dove 2001).

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 capabilities that are required to practice agility are, therefore, response ability and knowledge management. Response ability is achieved through change proficiency and flexible relationships that are reusable, reconfigurable, and scalable. Knowledge management in turn requires collaborative learning and knowledge portfolio management including the identification, acquisition, diffusion, and renewal of the knowledge the organization requires strategically (Dove 2001).

Haeckel's (1995, 1999) approach to adaptive organization implies radically different forms of governance, institutionalization of new norms of adaptive behavior, and translation of the organization's mission and practices into information that can easily be communicated and interpreted amongst its constituents. There are four basic principles for the adaptive organization. First, the traditional command and control approach that works well in stable, predictable environments is replaced. Instead, organization-specific governance mechanisms are adopted that support a high level of local autonomy, facilitate coordination across individual units, and empower individuals, groups, and organizational units to act on local knowledge while at the same time ensuring a sufficient level of coherent behavior overall. Second, procedures must be supplemented with personal accountability so each business process has two dimensions (Scherr 1993): procedure ("who or what does what to what and with what") and accountability ("who owes what to whom and by when") (Haeckel 1995, p. 11). Third, organizations must implement learning processes on different levels and within different

areas. These processes are based on recurrent sense-and-respond cycles (sense–interpret–decide–act). Fourth, the organization must be structured into modular processes. To be highly adaptive, an organization "has to have the ability to snap together modular processes and products as if they were Lego building blocks" (Haeckel 1995, p. 42).

The speed of change has increased within the software industry over the last decade (Baskerville et al. 2001; Holmberg and Mathiassen 2001) and there is consequently a need to adopt agile practices (Börjesson and Mathiassen 2004b). One comprehensive attempt to do so is expressed 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 agility mindset expressed here and in many related agile methods (Abrahamson et al. 2002) is, however, entirely focused on software development on the project level. Agility, though, is an organization-level capability. The challenge is to adopt agility practices broadly in response to unpredictable changes (Börjesson and Mathiassen 2004b). Such efforts must adopt software development agility and SPI agility as integral parts of building agile software organizations.

2.3 Navigating SPI Projects

In order to understand the need for, and approaches to, developing agile practices within SPI, we distinguish between two complementary tactics. The first, the supertanker tactic, is based on the CMM mindset, whereas the other, the motorboat tactic, is based on the agile mindset.

The supertanker tactic is centralist and top-down: (1) understand the current status of the process relative to CMM (or some other model), (2) create a vision of the desired process relative to CMM, (3) establish a list of required actions, (4) produce a plan for accomplishing these actions, (5) commit resources required to execute, and (6) start over at step 1 (Humphrey 1989, p. 4). Each loop contributes to the standards enforced in the organization.

Humphrey discusses the dilemma of process definition between project-unique circumstances on the one side, and the need for organization-level standardization in order to promote learning, teaching, measuring, and tool support on the other. This dilemma, he suggests, can be solved through tailoring standard software processes to suit specific project needs (p. 248).

To Humphrey, SPI targets a generic process that can be modelled, controlled, measured, and improved. Improvements will only stick if reinforced by careful introduction combined with periodical monitoring:

> The actions of even the best-intentioned professionals must be tracked, reviewed, and checked, or process deviations will occur. If there is no system

> to identify deviations and to reinforce the defined process, small digressions will accumulate and degrade it beyond recognition. (p. 22)

In order to achieve organization-level standardization, the supertanker tactic relies on SPI committees or specialist staff functions. This means a separation of thinkers from doers with a main focus on organization-wide processes. Key to this tactic is the process group—a collection of specialists that facilitate the definition, maintenance, and improvement of processes used by the organization (CMMI Product Team 2002a; 2002b).

The supertanker tactic is, in summary, centralist and top-down with an ambition to build and enforce organization-level standard processes, with process tailoring as the response to particular project conditions, with strict process enforcement, and with separation of thinkers from doers. The SPI governance structure emphasizes overall coordination of projects and organization-wide institutionalization of standard processes. Learning primarily takes place on the overall SPI level through continuous sense-and-respond cycles that identify current weaknesses, initiate new efforts, and implement their results organization-wide. In many ways this tactic relates to what Weick and Quinn (1999) call episodic change.

The problem with the supertanker tactic is its focus on procedure (i.e., on the tasks and decisions involved in producing software according to requirements). The focus on predictability and planning tends to render the software process bureaucratic based on command and control thinking.

The motorboat tactic is a critique of the CMM mindset and its deployment of solitary processes (i.e., one process is provided for any particular task). Kondo (2000) argues that solitary processes might lead people to feel that they are not responsible for nonconformance of product quality. Centralism thus might lead to software engineer alienation. Separating thinkers from doers means a split between insiders and outsiders in software projects leading to stifled motivation at both ends and to turf guarding if careers collide in a clash between project interests and general interests in staff functions or committees. Finally, process perfection will likely be static and fail to seize opportunities for improvement. As Kondo puts it,

> since standardized working means and methods have been formulated after careful consideration of all the angles, they must be the most productive and efficient means and methods possible, regardless of who uses them—at least the people who drew up the standards think so. (p. 9)

The motorboat tactic assumes that one size does *not* fit all. Processes are cultivated taking particular circumstances into account. The tactic includes strong elements of practice pull from software engineers (Börjesson and Mathiassen 2004c; Zmud 1984) and change agents work in specific projects to help software engineers help themselves.

Motorboat tactics develop software processes by combining best practices with experiences shared through networking (Aaen 2002). Processes are emergent and their use promotes discretion and latitude at the individual and project level in response to specific circumstances. As in agile approaches such as XP (Beck 2000), software processes serve primarily as a departure point for defining a common mode of operation

Table 1. Characteristics of the Supertanker and Motorboat Tactics

Issue	Supertanker	Motorboat
Organization	Centralist, top-down	Decentralist, bottom-up
Coordination	Between SPI projects	Between SPI and practice
Process	Generic	Dedicated
Diffusion	Process-push	Practice-pull
Learning	Software organization level	Software project level

in the team. Thinkers and doers work closely together as process experts serve as mentors, coaches, and consultants for a software team—or, they may even join the team for the duration of the project (Kautz et al. 2001).

The motorboat tactic is, in summary, decentralist and bottom-up with an emphasis on project and team level standardization of processes, with a focus on emergent processes that promote discretion and latitude at the level of the engineer, with an emphasis on practice pull to cultivate processes in response to situational opportunities, and relying on facilitation by mentoring and coaching software engineers by embedding change agents directly into software projects. Key to the motorboat tactic is the support of adaptive SPI practices. The governance structure emphasizes coordination between SPI experts and software engineers allowing for local authority in the project. Learning primarily takes place within projects through continuous sense-and-respond cycles that identify current weaknesses, initiate new efforts, and implement these as the project evolves and delivers its results. Thus this tactic relates to organizational change via improvisation (Orlikowski 1996) and to what Weick and Quinn label continuous change.

The problem with the motorboat tactic is that it tends to ignore organization-wide procedures. A sole focus on short-term challenges tends to focus on commitments and to render the software process *ad hoc*. The characteristics of the supertanker and motorboat tactics are shown in Table 1.

3 RESEARCH METHOD

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.

The authors represent industry and academia in close cooperation to secure relevant data and an appropriate theoretical framing of the study. The authors represent the insider and outsider perspective (Bartunek and Louis 1996). The insider is more connected to the problems than the outsider, while the outsider is more capable of unbiased reflection on the problem. The overall purpose of the research collaboration was twofold. We wanted to improve the understanding of how different SPI tactics at Ericsson affect the outcome, while at the same time contributing to the body of knowledge on SPI. We focused on how different SPI tactics affected the outcome of 18 different SPI projects. Collaborative practice research (Mathiassen 2002) was used to

Table 2. Data Sources

Data Source	Description
Direct involvement	One author was directly involved in 6 of the SPI initiatives and responsible for the remaining 12 initiatives
Participatory observations	One author participated in discussions of problems and results during SPI steering group meetings, training occasions, and informal meetings with software practitioners
SPI project documentation	Project meeting protocols, project presentations, project plans, project decisions, final reports
Minutes of meetings	Formal notes on discussions and decisions made about introduction of the new requirements management approach
Software development tools	Access to the software tools and their databases to understand the actual use of them
SPI survey	A yearly conducted SPI survey made by the SPI unit
Open-ended semi-structured interviews	Informal interviews were made with 10 practitioners to understand and validate the outcome of the initiative

frame this interpretive case study (Yin 1994). As one of the authors has been responsible for and active in the SPI initiatives, this study could also be viewed as action research (Baskerville and Pries-Heje 1999; Galliers 1992; Walsham 1995).

Data were collected and analyzed using a multi-method approach to establish a valid basis for analysis and to develop a thick description of the case (Mingers 2001; Yin 1994). Table 2 summarizes the list of data sources for our study. We used the different data sources to triangulate findings (Yin 1994).

4 THE CASE

Growing from 150 employees in 1995 to 900, in 2001 this particular Ericsson organization was reorganized and downsized during 2002 and 2003 to 550 employees. During this dynamic period, different SPI tactics were used to address and improve software development. The 18 different SPI initiatives were explored from different views as summarized in the Appendix (adapted from Börjesson and Mathiassen 2004c). In retrospect we can divide the 18 initiatives into two groups: The first 11 initiatives executed between 1998 and 1999 employed a supertanker tactic while the remaining 7 initiatives executed between 2000 and 2002 employed a motorboat tactic. We use the following attributes to distinguish between tactics and to understand how they affected outcomes:

- Improvement Set-up: How is the initiative organized and coordinated?

- Process Creation: On what basis and with what intentions are processes created?
- Process Diffusion: How are new processes diffused into practice?
- Navigation Ability: What kind of learning takes place in order to sense and respond to changes in the environment?

Our focus is on implementation success—the actual changes in software engineering practice. Implementation success is directly recognizable, unlike SPI success, and although not synonymous with SPI success, implementation success is a prerequisite for SPI success. The next three subsections explore and compare the two tactics.

4.1 The Supertanker

The 11 supertanker initiatives had low or medium implementation success (Börjesson and Mathiassen 2004c). The initiatives targeted key practice areas such as configuration management, requirements management, project tracking, module testing, and subcontract management. Table 3 shows how they were managed.

Table 3. Features of the SPI Supertanker Tactic

Improvement Set-up	*One size fits all*: An organization benefits from centralist SPI initiatives. All product units within the organization can streamline their practices by having one improvement initiative serve common needs. Special interests are served through adaptation and tailoring. Having several product units benefit from one SPI initiative is cost effective.
Process Creation	*High focus on representation in the initiatives*: All SPI decisions are taken in an SPI steering group consisting of representative managers from each unit. The steering group discusses, decides, ensures resources, and follows up on the initiatives. No special attention is devoted to process implementation and use. There is a high focus on good representation from all units. Steering group members prepare and attend steering group meetings and follow up on action points.
Process Diffusion	*Based on the CMM Mindset*: The SPI work is organized through a software engineering process group (SEPG) (Fowler and Rifkin 1990; Humphrey 1989). The SPI group works on a corporate level to support all product units in the organization. CMM is used to assess the organization (identified as level one in 1999) and CMM Light assessments are implemented in ongoing software engineering projects.
Navigation Ability	*High focus on initial requirements*: Serving common needs among product units implies a need to hold on to initial requirements as changes will require negotiations with every stakeholder involved.

In general, supertanker initiatives had high focus on defining a solution, but very little on deployment of the process. For instance, understanding how resistance to change could make SPI initiatives fail received little attention (McFeeley 1996). Serving all product units and their different needs simultaneously made the SPI initiatives spend most of their efforts in the early phases of the improvement work (Börjesson and Mathiassen 2004c). As a result, there was limited interaction between the SPI initiatives and the development projects (except via the representatives participating in the initiative). Interaction was mainly through one-way communication from initiative to software engineers. All initiatives using the supertanker tactic were financed at the organizational level. The development projects did not pay for the solutions they received from the SPI initiatives.

4.2 The Motorboat

The seven motorboat initiatives had high implementation success (Börjesson and Mathiassen 2004c). As with the SPI supertanker initiatives, the motorboat initiatives targeted key practice areas such as requirements management, test, configuration management, and project management. Table 4 shows how the initiatives in the motorboat approach were managed.

In general, there was a high focus on collaboration between the SPI initiative and the software engineers (i.e., the development projects). The initiatives targeted projects as well as concrete activities within projects. The initiatives supported actual work, if necessary by devising new solutions such as editing product requirements into a new database to ensure implementation and use. The initiatives were organized as sub-projects within the development projects to ensure close cooperation with end users. Both the SPI initiative and the main engineering project manager in question had the double goal of improving practice and managing the engineering project time schedule. Contacts between the initiative and software engineers were on a daily basis and together they negotiated new requirements and ideas as they came up. The engineering projects financed the SPI initiative. When more than one engineering project was expected to use the result, the main engineering project covered the costs.

4.3 Similarities and Differences

The two tactics had many similarities. Several SPI change agents, managers, and practitioners took part in executing both tactics. Management attention, involvement, and commitment were also high in both cases. The types of products (embedded real-time systems) and improvement areas (requirements management, configuration management, etc.) were the same. In both tactics there was one SPI group responsible for driving the initiatives.

As the outcomes differ between the two tactics, a number of differences stand out. Except for the four attributes we have studied in the previous sections, there are several general differences that may have impacted the outcome. In the motorboat tactic, the SPI change agents worked with only one or a few practices and related tools. In the

Table 4. Features of the SPI Motorboat Tactic

Improvement Set-up	*One size fits one*: A product unit benefits from dedicated SPI initiatives. SPI initiatives are aligned to the extent that the product unit in question works with other organizational units on the same product. Specifically, the dedicated SPI initiative focuses on one engineering project to ensure implementation and use. The initiative focuses on just in time solutions for the engineering projects. Simplicity and relevance makes implementation and use cost effective.
Process Creation	*High focus on implementation and use*: All SPI decisions are taken in an SPI steering group consisting of representative managers from each practice (requirements, design, test, etc.) within the product unit. The steering group discusses, decides, secures resources, and follows up on the initiatives. High attention is devoted to process implementation and use. There is a high focus on having good representation from all practices in the steering group. The project manager is a member of the steering group. Steering group members prepare and attend steering group meetings and to follow up on action points.
Process Diffusion	*Ad hoc mindset:* No special attention is put on any established SPI theory. Serving specific and recognized needs in the product unit are believed to lead to implementation success.
Navigation Ability	*High focus on deployment*: Serving specific and recognized needs in the product unit implies that changing needs can be accommodated subject to particular demands from other organizational units working on the same product.

supertanker approach, the SPI change agents had a wider competence base, but not a very deep one. For SPI change agents, it was easier to collaborate with software engineers when acting as specialists within a process area. When using the supertanker tactic, the change agent role received little attention from management and the SPI group itself. Neither change management nor diffusion of innovations models—the Satir change model (Weinberg 1997) and the diffusion of innovation curve (Rogers 2003)—were known or discussed. When using the motorboat approach, these areas had been identified as important factors for SPI implementation. Management commitment was high in both tactics; however, there were differences in the level of commitment. In both tactics, software managers believed in the value of the improvement work, but only in the motorboat tactic was their attention prominent. In the motorboat tactic, managers joined the SPI steering group, as they knew that if managed sensibly they could really benefit from the initiatives. Conversely, in the supertanker approach, every manager in the SPI steering group knew from the outset that it would be difficult to make the initiative cater for their special needs with so many stakeholders involved.

5 DISCUSSION

During the last two decades, diffusion of technology innovations has increased enormously. Computers in every home and on each desk, mobile phones, Internet connections, and wireless communication are today a fact. When the traditional CMM mindset was in its cradle, a lot of this was hard to imagine. These changes have made the world more maneuverable and things tend to happen much faster as a result. The traditional CMM mindset, which might be efficient in a less-changing environment, has become a supertanker. CMM's basic ideas are still very much the same, but the environment in which the model is used has changed. There are still values in having shared languages, a common ground for learning, and corporate processes for economic reasons. The traditional CMM mindset is, however, no longer ideal to accomplish these values in an ever-changing business environment.

None of the initiatives using the supertanker tactic were successful in implementing and using new practices. Börjesson and Mathiassen (2004c) point out a number of reasons: low process-push and practice-pull, little iteration, little effort spent in the deployment phases, and no consideration of resistance to change. Börjesson and Mathiassen (2004a) offer a more elaborate analysis of process-push and practice-pull issues related to this case study. The CMM mindset of one-size-fits-all does not help SPI change agents to focus on software engineering needs and practices. Too many stakeholders are involved to attain implementation success via high process-push. Such tactics might work well in stable environments, but in our cases there were substantial organizational and project dynamics such as growth followed by downsizing, organizational changes, and numerous project-specific changes of importance for the SPI initiatives. Together, these factors render SPI initiatives based on the CMM mindset into a supertanker that cannot navigate in muddy waters.

The motorboat tactic described in Table 4 was not based on any special theory. The general belief was that a focus on one-size-fits-one would make it possible to set up improvement initiatives where SPI change agents could focus on software engineering needs to assure successful implementation and use. The constantly incoming change requests could be managed, as there were a limited number of stakeholders and, therefore, room for the SPI change agents to sense the ongoing action and respond to changes. The motorboat tactic made it possible to navigate in muddy waters.

The motorboat strategy is a bottom-up approach to improvement. It does, however, not help much in coordinating at the corporate level. One could hope that as people move to new departments and projects, they will bring along their successful processes to new uses and contexts and thereby diffuse new practices. The use of bottom-up strategies may, however, lead to balkanization, confusion, and loss of good experiences in politicized surroundings at the corporate level. Likewise there is no guarantee that bottom-up SPI will result in anything coherent at the corporate level within reasonable time. Great ideas at the local level may not compute at the global level.

Haeckel (1995) offers an elegant solution to the possible conflict between the local and the global. Where the CMM mindset assumes that processes at the global and local level are modeled using similar language (i.e., modeling the local process as a specialization of a global process), the agility mindset lends itself to separate thinking at the two levels. At the local level—the team or project level—procedures and methods are

instrumental in negotiating and using common practices. At the global level—the corporate or organizational level—we can adopt Haeckel's ideas, pointing to values, governing principles, and governance models as instrumental in building a common ground for local and global improvement efforts and practices.

Such an agile strategy combines reactive and proactive change proficiency in building efficient and adaptive development processes. It is always important to include efficiency into thinking about SPI practices. In a changing environment, it is, however, as we argue in this paper, equally important to introduce and practice an adaptive mindset. Haeckel (1995) offers governance mechanisms that support enterprise design without leading to command and control structures, but still building corporate coherence in action.

Further research is needed to understand potential combinations of volume, push, pull, target, success, tactic, complexity, etc. for SPI initiatives. There is, of course, a possibility that project volume or combination of push and pull have a greater impact on the result than what is foreseen in this study. Based on available data and the framework used for analysis, our results indicate that SPI tactics significantly affect SPI implementation success, and that the motorboat tactic has led to higher implementation success than the supertanker tactic.

6 CONCLUSION

This study explores and analyzes 18 different SPI initiatives within the telecom company Ericsson in Gothenburg, Sweden. Four attributes— (1) improvement set-up, (2) process creation, (3) process diffusion, and (4) navigation ability—help us identify two different SPI tactics, the motorboat and the supertanker. The supertanker tactic follows a CMM mindset, while the motorboat tactic builds on agile ideas. There is no recognized SPI model supporting the motorboat tactic and this study identifies a need for such. We suggest that a combination of motorboat tactics and modified supertanker tactics is key to successful SPI. We need more research—empirical as well as theoretical—investigating how to build SPI agility by combining bottom-up improvements into coherent practices at the corporate level. Specifically, we need to further investigate *how* adaptive governance approaches and principles on the corporate level can ensure appropriate levels of overall coherence and efficiency.

REFERENCES

Aaen, I. "Challenging Software Process Improvement by Design," in *Proceedings of 10th European Conference on Information Systems* (Volume 1), S. Wrycza (Ed.), Gdansk, Poland, 2002, pp. 379-390.

Abrahamson, P., Salo, O., Ronkainen, J., and Warsta, J. *Agile Software Development Methods: Review and Analysis*, Oulu, Finland: VTT Electronics: Publication #478, 2002 (available online through http://www.inf.vtt.fi/pdf/).

Bartunek, J. M., and Louis, M. R. *Insider/Outsider Team Research*, Thousand Oaks, CA: Sage Publications, 1996.

Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B., and Slaughter, S. "How Internet Software Companies Negotiate Quality," *Computer* (34:5), 2001, pp. 51-57.

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

Beck, K. *Extreme Programming Explained: Embrace Change*, Reading, MA: Addison-Wesley, 2000.

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. "Manifesto for Agile Software Development," 2001 (available online at http://www.agilemanifesto.org).

Börjesson, A., and Mathiassen, L. "Making SPI Happen: The Road to Process Implementation," in *Proceedings of the 12th European Conference on Information Systems*, T. Leino, T. Saarinen, and S. Klein (Eds.), Turku School of Economics and Business Administration, Turku, Finland, 2004a.

Börjesson, A., and Mathiassen, L. "Organization Dynamics in Software Process Improvement: The Agility Challenge," in *IT Innovation for Adaptablity and Competitiveness*, B. Fitzgerald and E. Wynn (Eds.), Boston: Kluwer Academic Publishers, 2004b, pp. 135-156.

Börjesson, A., and Mathiassen, L./ "Successful Process Implementation," *IEEE Software* (21:4), 2004c, pp. 36-44.

CMMI Product Team. *CMMI for Software Engineering, Version 1.1, Continuous Representation (CMMI-SW, V1.1, Continuous)*, Pittsburgh, PA: Software Engineering Institute Technical Report CMU/SEI-2002-TR-028,), pp. 2002a (available online at http://www.sei.cmu.edu/publications/documents/02.reports/02tr028.html).

CMMI Product Team. *CMMI for Software Engineering, Version 1.1, Staged Representation (CMMI-SW, V1.1, Staged)*, Pittsburgh, PA: Software Engineering Institute Technical Report CMU/SEI-2002-TR-029, 2002b (available online at http://www.sei.cmu.edu/publications/documents/02.reports/02tr029.html).

Creech, B. *The Five Pillars of TQM: How to Make Total Quality Management Work for You,* New York: Truman Talley Books/Dutton, 1994.

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

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

Feiler, P. H., and Humphrey, W. S. *Software Process Development and Enactment: Concepts and Definitions*, Pittsburgh, PA, Software Engineering Institute, 1991.

Fowler, P., and Rifkin, S. *Software Engineering Process Group Guide*, Pittsburgh, PA: Software Engineering Institute, 1990.

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

Grady, R. B. *Successful Software Process Improvement*, Upper Saddle River, NJ: Prentice Hall, 1997.

Gunneson, A. O. *Transitioning to Agility: Creating the 21st Century Enterprise*, Reading, MA: Addison-Wesley, 1997.

Haeckel, S. H. *Adaptive Enterprise: Creating and Leading Sense-and-Respond Organizations.* Boston: Harvard Business School Press, 1999.

Haeckel, S. H. "Adaptive Enterprise Design: The Sense-and-Respond Model," *Planning Review* (23:3), 1995, pp. 6-13, 42.

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

Humphrey, W. S. *Managing the Software Process*, Reading, MA: Addison-Wesley Publishing Company, 1989.

Kautz, K., Hansen, H. W., and Thaysen, K. "Understanding and Changing Software Organizations: An Exploration of Four Perspectives on Software Process Improvement," *Scandinavian Journal of Information Systems* (13), 2001, pp. 31-50.

Kondo, Y. "Innovation Versus Standardization," *TQM Magazine* (12:1), 2000, pp. 6-10.

Mathiassen, L. "Collaborative Practice Research," *Information, Technology & People* (15:4), 2002, pp. 321-334.

Mathiassen, L., Pries-Heje, J., and Ngwenyama, O. *Improving Software Organizations*, Boston: Addison-Wesley, 2002.

McFeeley, B. *IDEAL: A User's Guide for Software Process Improvement*, Pittsburgh, PA: Software Engineering Institute Handbook CMU/SEI-96-HB-001, 1996 (available online at http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.001.html).

Mingers, J. "Combining IS Research Methods: Towards a Pluralist Methodology," *Information Systems Research* (12:3), 2001, pp. 240-259.

Orlikowski, W. J. "Improvising Organizational Transformation Over Time: A Situated Change Perspective," *Information Systems Research* (7:1), 1996, pp. 63-92.

Osterweil, L. J. "Software Processes Are Software Too," in *Proceedings of the 9th International Conference on Software Engineering*, Los Alamitos, CA: IEEE Computer Society Press, 1987.

Osterweil, L. J. "Software Processes Are Software Too, Revisited," in *Proceedings of the 19th International Conference on Software Engineering*, Los Alamitos: IEEE Computer Society Press, 1997.

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

Paulk, M. C., Weber, C. V. , Curtis, B., and Chrissis, M. B. *The Capability Maturity Model: Guidelines for Improving the Software Process*, Reading, MA: Addison-Wesley Publishing Company, 1995.

Paulk, M. C., Weber, C. V., Garcia, S. M., Chrissis, M., and Bush, M. *Key Practices of the Capability Maturity Model, Version 1.1*, Pittsburgh, PA: Software Engineering Institute, 1993.

Pries-Heje, J., and Tryde, S. "Diffusion and Adoption of IT Products and Processes in a Danish Bank," in *Diffusing Software Products and Process Innovations*, M. A. Ardis and B. L. Marcolin (Eds.), Boston: Kluwer Academic Publishers, 2001, pp. 17-34.

Rogers, E. M. *Diffusion of Innovations*, New York: The Free Press, 2003.

Scherr, A. L. "A New Approach to Business Processes," *IBM Systems Journal* (32:1), 1993, pp. 80-98.

Walsham, G. "Interpretive Case-Studies in IS Research: Nature and Method," *European Journal of Information Systems* (4:2), 1995, pp. 74-81.

Weick, K. E., and Quinn, R. E. "Organizational Change and Development," *Annual Review of Psychology* (50), 1999, 361-86.

Weinberg, G. M. *Quality Software Management: Volume 4, Anticipating Change*, New York: Dorset House Publishing, 1997

Yin, R. K. *Case Study Research: Design and Methods*, Thousand Oaks, CA: Sage Publications, 1994

Zmud, R. W. "An Examination of 'Push-Pull' Theory Applied to Process Innovation in Knowledge Work," *Management Science* (30:6), 1984, pp. 727–738.

ABOUT THE AUTHORS

Ivan Aaen is an associate professor of Computer Science at Aalborg University, Denmark. His interests include software engineering and information systems. He holds a Ph.D. in Computer Science from Aalborg University and is a member of the ACM and IEEE Computer Society. Ivan can be reached at aaen@acm.org.

Anna Börjesson is a software process improvement manager at Ericsson AB in Gothenburg, Sweden, and an industrial Ph.D. student at the IT University in Gothenburg. She has over 10 years of software engineering working experience. More than 7 of those years have been dedicated to SPI, change management, and diffusion of innovations. Anna is a member of IEEE and ACM. She can be reached at anna.borjesson@ericsson.com.

Lars Mathiassen is Georgia Research Alliance Eminent Scholar and Professor in Computer Information Systems at Georgia State University. His research interests focus on engineering and management of IT systems. In particular, he has worked with project management, object-orientation, organizational development, management of IT, and the philosophy of computing. Lars can be reached at lmathiassen@gsu.edu.

APPENDIX[1]

#	Improvement Area	Volume	Target	Process Push	Practice Pull	Implementation Success	SPI Tactic
1	Configuration Management	10 weeks 300 person hours 4 participants	Several units	Weak	Weak	Low. The result was considered hard to use. Part of the knowledge gained was used indirectly in other projects.	Supertanker
2	Design Information	21 weeks 400 person hours 6 participants	Several units	Weak	Weak	Medium/Low. The intention was to provide a framework for design. The results were mainly implemented in one project.	Supertanker
3	Estimation and Planning	14 weeks 600 person hours 11 participants	Several units	Weak	Medium	Low. The results were tried out in one project, but it ran into difficulties. Support was weak, and no one helped the project.	Supertanker
4	Historical Data	16 weeks 200 person hours 4 participants	Several units	Weak	Weak	Low. The purpose was to build a database of old data and take action from there, but no interesting data were found.	Supertanker
5	Introductory training	14 weeks 620 person hours 11 participants	Several units	Weak	Strong	Medium. An estimated 50% of managers used the process. Some did not know about it and were not given the opportunity to learn about it.	Supertanker
6	Module Test	12 weeks 400 person hours 10 participants	Several units	Weak	Weak	Medium/Low. The process was used when process engineers were members of a project or when section managers strongly believed in systematic module tests.	Supertanker
7	Project Tracking	9 weeks 300 person hours 7 participants	Several units	Weak	Weak	Medium/Low. The process was used in one project supported by the driver of the SPI initiative.	Supertanker

[1]The table in this Appendix is adapted from Börjesson and Mathiassen (2004c).

#	Improvement Area	Volume	Target	Process Push	Practice Pull	Implementation Success	SPI Tactic
8	Resource Handling	4 weeks 250 person hours 8 participants	Several units	Weak	Medium	Medium. An estimated 75% of managers used the new process. Non-users either did not get the help they needed or did not believe in the approach.	Supertanker
9	Requirements Management	10 weeks 200 person hours 5 participants	Several units	Weak	Weak	Low. The results were hard to use. The members of the SPI initiative mainly used the results as a framework.	Supertanker
10	Requirements Management Implementation	12 weeks 330 person hours 7 participants	Project	Strong	Weak	Medium. The initiative was started because of the low impact of initiative #9. The focus was on one project to suit its needs.	Supertanker
11	Subcontract Management	18 weeks 650 person hours 9 participants	Several units	Weak	Weak	Low. The results needed further adaptation to be useful for different projects, but no effort was made.	Supertanker
12	Requirements Management	30 weeks 1,200 person hours 3 participants	Project	Strong	Strong	High. The process was adapted to a specific project, but needed further adaptation. Process engineers and practitioners solved these problems jointly.	Motorboat
13	Analysis & Design	30 weeks 1,000 person hours 4 participants	Unit	Strong	Strong	Medium/High. This complex area required several iterations of experimenting with processes before the result was satisfactory.	Motorboat
14	Implementation	30 weeks 1,000 person hours 4 participants	Project	Strong	Strong	High. Two slightly different adaptations were made to fit the needs of different products developed on different sites.	Motorboat
15	Test	30 weeks 1,300 person hours 2 participants	Unit	Strong	Strong	High. The result was adapted to a specific project. Process engineers and practitioners solved difficulties together.	Motorboat

#	Improvement Area	Volume	Target	Process Push	Practice Pull	Implementation Success	SPI Tactic
16	Configuration Management	30 weeks 1,650 person hours 6 participants	Unit	Strong	Strong	High. There are many possible solutions for each specific situation. Attention was required to choose one. The dedication of the practitioners was key.	Motorboat
17	Project Management	10 weeks 150 person hours 2 participants	Unit	Strong	Strong	High. The results were used and the project managers' dedication to SPI within project management continued.	Motorboat
18	Process Development Map	30 weeks 200 person hours 2 participants	Unit	Strong	Strong	High. The use of the map is measured in both "hits" and subjective opinions of the need. Measurements are very positive.	Motorboat