Combining Agile Methods with Stage-Gate Project Management

Daniel Karlström and Per Runeson, *Lund University*

Case studies at three large, software product companies show the benefits and pitfalls of integrating agile methods with stage-gate management models.

The project manager enters the room where the pilot agile development team sits. "We have to add a new feature to our next release," he says. "Can you please put it on the list?"

"Yes, of course," a team member replies. "Just let us know which feature you want us to postpone in its place."

The manager gets confused and a little annoyed. "What's happening? When we used our traditional development methodology, you could squeeze in all the new features we needed. I feel as though I'm no longer in control here."

This fictitious dialogue captures the cultural changes and different management and engineering viewpoints we encountered in a study of three large software development organizations that started using agile methods.

We undertook the study in response to questions raised at an industrial seminar on Extreme Programming (XP)[1,2] and agile development methods held in 2003. The Software Engineering and Research Group at Lund University sponsored the seminar, which included representatives from more than 20 different software companies. The representatives were extremely interested in agile development, but they weren't sure whether or how it might coexist with their existing project management models.

Their interest prompted us to study the feasibility of applying agile methods in the context of large software development projects using stage-gate project management models (see the related sidebar). The results showed not only the feasibility but also the benefits of this approach as well as issues that must be addressed to ensure its success.

Case studies' scope

The software engineering community has generally viewed agile methods favorably. The focus on simplicity, programmers, and products empowers the technical staff and gives a back-to-basics sense to their work. This contrasts starkly with process-oriented and maturity-oriented approaches, such as CMM, which empower management and quality departments by adding procedures and standards to improve products. All the engineering teams in the software development companies

0740-7459/05/$20.00 © 2005 IEEE

Stage-Gate Project Management Models

A stage-gate model describes a work process from idea to delivered product. The model presents the stages of product development in a generic, abstract project life cycle suitable for high-level management.

Figure A shows the five stages of the classic Cooper stage-gate model.[1] Other models have other labels but the same principles.[2] Two stages are connected through a gate. At each gate, top management and the project sponsor decide on whether to continue funding the project and move to the next stage.

At a more detailed level, a stage-gate model is connected to a technical engineering process model, such as a waterfall development process or an iterative software development process. Milestones in the engineering process are connected to gates in the stage-gate model.

References

1. R.G. Cooper, *Winning at New Products*, 3rd ed., Perseus Publishing, 2001.
2. L. Mulder, "The Importance of a Common Project Management Method in the Corporate Environment," *R&D Management*, vol. 27, no. 3, 1997, pp. 189–196.

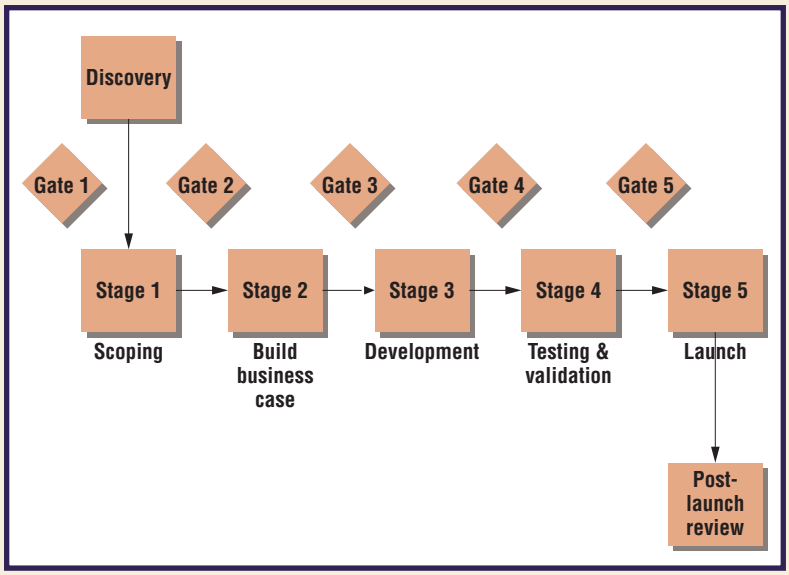

Figure A. Cooper stage-gate project management model.

we studied were enthusiastic for making process changes to introduce agile methods.

The company management in all cases was convinced of the need for change; the question was what to change to and how to implement change in the face of higher-priority delivery deadlines.

As a first step in our study, we reviewed the literature and found a mismatch between the companies' concerns and the available information. The companies run in global, product-driven markets, using stage-gate models to manage development projects. By contrast, agile methods are derived from small-scale, contract-driven software development projects. Much of the agile literature focuses on a contract situation over a product situation. Moreover, excepting some preliminary research by Christina Wallin, Fredrik Ekdahl, and Stig Larsson,[3] little research is available on integrating agile into stage-gate models, and the authors of a second, more recent study note that data collection for the research was less than ideal.[4] Our research objective therefore focused on investigating the integration of agile teams into stage-gate software product development, adhering to a strict qualitative research process.

Managing single-iteration contracts differs considerably from managing for software products or product lines, which a company maintains continuously. Traditional projects plan and allocate resources completely before work begins. The development stages—requirements, high-level design, detailed design, testing, and "productizing"—are either sequential or iterated in long cycles. The process tries to minimize requirements changes so that the product is completed on time. Functionality is the fixed variable in traditional projects.

In contrast, agile methods try to minimize the effects of changes at any time in the product life cycle. Time is the fixed variable in agile projects.

Two of the companies we studied—ABB Automation and Ericsson Microwave Systems—were trying to use XP in limited parts of a software development project that was governed by their respective stage-gate management models. In both cases, the companies changed the software engineering methodology without changing the stage-gate model. However, the companies had quite different introduction strategies and politics. Our studies aimed primarily to gather experience data for use in improving the stage-gate and technical engineering models.

At Vodafone, we studied the stage-gate model itself from a program and product management perspective. Our goal was to identify issues that might benefit from including agile influences in the stage-gate model at this level.

Companies and products

Table 1 summarizes some key characteristics of the three organizations we studied: ABB Automation, Ericsson Microwave Systems, and Vodafone Group global product develop-

Table 1

Overview of the cases involved in the study

Company	Product type	Customer type	Business model	Software engineering methodology	SEPM* model	Goal
ABB	Real-time industrial control and automation system	Industrial customers	Market	Normally traditional; pilot team using XP	Stage-gate	Agile software team integration into gate model
Ericsson	Embedded radar control and target tracking	Military	Product line with contracts	Normally traditional with some iteration, pilot team using XP	Stage-gate, iterative	Agile software team integration into gate model
Vodafone	Complex hardware-software system of systems	Consumer mass market	Market	Traditional	Stage-gate	Agile influences in gate model

*Software Engineering Project Management

ment. These organizations have some key characteristics in common: they are all branches of large global companies with several thousand employees and worldwide markets, and they all use corporate stage-gate project management models for product development.

However, their application domains, products, and markets differ substantially. ABB operates in the industrial automation domain, providing automation systems to the industry market. Ericsson Microwave operates in the defense domain, constructing radar surveillance and target tracking systems on a contract basis with military customers. Vodafone Group provides telecom services and products to a global mass market of approximately 140 million customers in 26 countries.

ABB Automation

A pilot XP project at ABB originated in a developer group's interest. The group was influenced by the general hype surrounding XP in software media. The organization didn't tailor its stage-gate models specifically for the agile projects, but management supported the initiative by providing resources for coaching software engineers. They authorized training expenses from the staff training budget and obtained financial support from the corporate research division as well.

The XP team delivered its product releases ahead of schedule and under budget. One or two team members were at first doubtful but soon turned around, and the team functioned together for about three years. It's currently dispersed to help other projects that are not using agile methods but are past their delivery deadline.

Ericsson Microwave Systems

The Ericsson initiative also came from a small engineering group that had become interested in agile methods from the media hype. In this case, however, the engineers started an internal study group on their own, approaching management later about introducing XP in their own software development work. Management's initial response was indifferent; the group could try it as long as it didn't interfere with the work to be performed.

The XP team received no support from their own management or the rest of the organization during the introduction of the methodology or the project's execution. The team developed the assigned functionality successfully, but were disbanded after a two-year pilot project. One reason is that management came to consider the methodology a threat during the project. However, it has since become very interested in the methods, because the XP-developed parts' quality is significantly higher than the quality of other parts of the project.

Vodafone Group

The Vodafone study perspective was different. The focus was on product management in the stage-gate model. Management had heard about agile product development from engineering and wanted to see which ideas could apply at a higher level in the company. The company is working on improving its product stage-gate process and is interested in agile methods as one source of inspiration.

The Vodafone product development environment is the fastest paced and most volatile of all three companies, owing to the large number of operators and countries involved as

Table 2

Findings summary for the effects of agile methods in three industry cases

Area	Agile feature	Effect*
Planning and prioritization	Most important feature first	+ Early feedback on features + No delays of important features
	Micro planning	+ Avoidance of requirements' cramming + Fixed plans avoided ! Little support for long-term plans
Communication and follow-up	Coherent teams	+ Good internal communication ! Potential isolation of agile team
	Automatic testing	+ Means for communication of change + Higher quality
	Small, manageable tasks Continuous integration	+ Feeling of being in control + Higher quality + Progress measure for management
Process model and roles	Customer involvement	+ Continuous feedback + Relevant features + Technical product manager is a good candidate as customer representative
	Documentation as tasks	+ Priorities resolved between documentation and code ! Conflicts visible between different amounts of documentation
Project management	Engineering-level empowerment	+ Engineers feel motivated ! Managers afraid initially ! Management training needed
	Focus	+ Engineers focus on past and current release, managers on current and future release + (!) Technical issues raised (too) early for management
	Engineering-level initiative	+ Little resistance to change

*+ Indicates positive effect, and ! indicates effects that might need attention.

well as the intense competition in the telecom market. Despite significant enthusiasm for change from both software engineering and marketing, change is proving difficult to prioritize over day-to-day deadlines. Other issues complicating the Vodafone case include the large number of people involved in a change made at higher organizational levels and the geographic distances involved in distributed development teams. As a result, the change initiative is moving slower than expected.

Research findings

We adopted a qualitative approach for this research, based on semistructured interviews and archival analyses in the three companies (see the "Research Methodology" sidebar). The analysis procedure resulted in findings in four areas, summarized in Table 2, where a "+" indicates positive effects and a "!" indicates effects that might require attention.

Planning and prioritization

In both the ABB and Ericsson cases, the agile development style implies that the engineering team is working on the most important features at any given time and completing them first. This establishes earlier feedback loops on the most important features and keeps the project deadline from affecting their scope. Only less important features might be scaled back or dropped.

Agile methods also eliminate the "requirements cramming" problem. In stage-gate development, management might ask engineering to squeeze in extra features now and then. Uncertainty about overall progress as well as how much work each plan increment required made it relatively easy to add such features without excluding something else. The work to be done would gradually increase, along with uncertainty about the project's progress.

XP microplanning lets developers set these requests up as trade-offs: "Sure, we can do that, but what do you want to take out?" In the beginning, this frustrated management in the Ericsson case, but later, it was considered a benefit.

The higher iteration frequency in agile development also provides more opportunities for product feedback. Replanning with Gantt charts and requirements specifications takes a lot of management time. This becomes more apparent as the business environment becomes more volatile and fast-paced. Planning a fixed schedule is difficult in these environments. In the Vodafone case, managers would sometimes hardly have time to complete change planning, let alone implementation. A strategy that facilitates change would be more effective.

Communication and follow-up

Both the ABB and Ericsson XP teams experienced improved communication among their members, but other teams perceived them as more isolated. It's positive for the team to get on with their work undisturbed, but negative if they're cut off completely from the larger organization. For example, resolving dependencies requires formal communication of some kind.

Any company attempting to combine agile with stage-gate methods must resolve communication issues. ABB often resolved dependencies through automatic testing of dependent functionality. Failed test cases indicated the occurrence of a change that had an impact on other features. Direct person-to-person communication proved more effective than document-based communication in resolving these matters, but management especially found this difficult to accept.

Because agile teams deliver actual functionality in running software, rather than documentation and partially developed functionality, they can easily demonstrate project status to customers. The demonstration also constitutes a tangible progress report in the stage-gate model. Management and developers perceived the project status much more clearly than with the traditional process, and the projects spent less time estimating progress overall.

The agile practice of splitting tasks into small, easily managed packages made it easy to focus. Teams experienced less confusion and expressed a sense of having their work "under control." This practice also all but eliminated the problem that requirements changes pose under traditional engineering processes, which require system replanning, redesign, and recoding. The teams also reported better understanding of the system's technical inner workings.

The quality of delivered product parts increased markedly. The teams themselves saw this improvement, as did other teams relying on the parts. Figure 1 shows the drop in defects reported for the ABB development team. (Note that the data for 2000 and 2004 is only for the part of the year during which the project was active—namely, fall 2000 and spring 2004.) Management attributes some of the quality improvement to the agile practice of

Research Methodology

We are convinced of quantitative research's effectiveness in software engineering process research,[1] but such methods are extremely difficult to apply to some problems. For example, qualitative methods are often more effective for investigating broad research questions.[2] The integration of agile teams into software-producing organizations is such a question, unlikely to yield to a laboratory setup or a purely quantitative approach in the actual context.

Figure B presents an overview of our methodology, though it simplifies the qualitative analysis, which occurs in several steps.

We conducted a total of 18 semistructured interviews at the three companies, approximately one hour each, with employees in different positions, including engineering, management, product management, and project management. The researchers recorded, transcribed, and analyzed the interviews. We used several techniques to increase the study's validity—for example, researchers performed certain steps of the analysis independently. The conclusions are fully traceable to individual interviewee statements.

Our research group is working closely with two of the companies in cooperative, long-term research, which implies tacit knowledge about the company operations and products.

References

1. C. Wohlin et al., *Introduction to Experimentation in Software Engineering*, Kluwer Academic Publishers, 1999.
2. C. Robson, *Real World Research*, Blackwell Publishers, 2002.

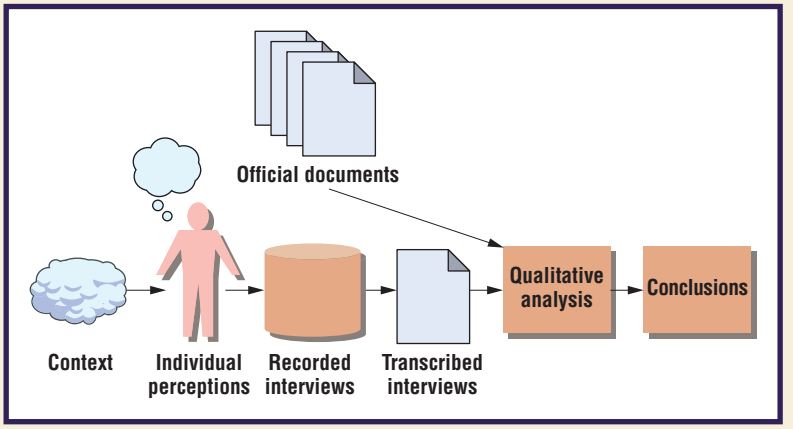

Figure B. Research methodology overview. The qualitative analysis actually occurs in several steps before drawing conclusions from the overall process.

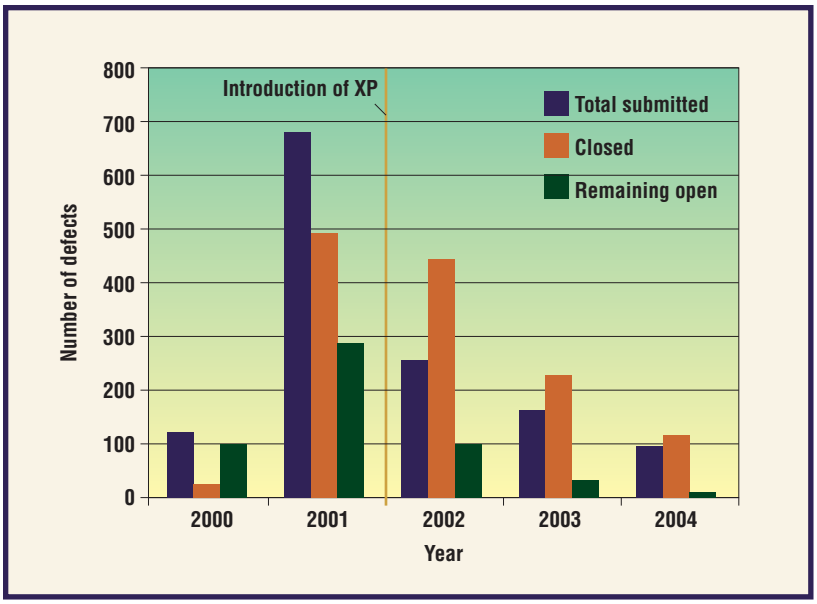

Figure 1. Defect report statistics from the ABB Automation development team.

continuous integration, which eliminates the "big bang" integration period that characterizes meeting delivery milestones in traditional processes.

Process model and roles

The agile requirement to identify a customer representative worked well with the stage-gate model. This was particularly true in the ABB case. Identifying a customer representative increased the amount and—equally important—the rate of feedback on the performed work. The technical product manager was the most natural person to assume this responsibility on a day-to-day basis, providing a single point of contact for both the team and senior management. Quick feedback eliminated erroneous functionality at an early development stage. The customer representative also kept the team conscious of someone directly in need of their work products.

The Ericsson team had more difficulty identifying a clear customer representative. Without this role in place, team members felt more as if they were developing software for the shelf.

The customer role is complex on these large projects, and it became apparent that the process required different customer representation at varying intervals—for example, customer representation for marketing product management, program management, usability testers, end users, and other software teams. At the project management level, it was important to have both a clear customer acting as a link to the developers as well as a clear overall leader and decision maker, coordinating marketing with technical and program management within the common stage-gate framework.

The development teams planned documentation work as tasks like any others. This proved very successful by clarifying that code development and documentation use the same resources. This made it easy to prioritize the tasks effectively. However, it also exposed the conflict between the large amount of documentation the stage-gate model requires and the lesser amount the engineers request.

Project management

In all three study cases, management acknowledged a need to change the traditional way the companies developed software products. The problem was that neither they nor the development teams really knew what changes to make. After the pilot studies, all the companies decided that the agile approach is a serious alternative and are considering ways it can influence future corporate methodologies and processes. Both management and developers agree that actual development doesn't adhere strictly to current models anyway. Especially under deadline pressures, development proceeds in an ad hoc, short-sighted way.

That the process change initiative came directly from engineering, instead of management, appears to be a success factor. The development team is highly motivated to accept the new methodology. The Ericsson team, however, encountered the problem of not having initial acceptance at the management level. Companies must take steps to increase support and understanding throughout the organization. They can do this by offering appropriate training at each company level and function. They might also establish forums where individuals can express their ideas.

At the product management level, the company has to maintain focus on the current product development and on its future direction. The development team is naturally focused on developing the current product. They must also fix the faults in recently developed products, but they are less concerned about future releases. Developers can easily perceive this difference in focus as a lack of interest in their work. Integrating agile methods with the stage-gate model keeps the development

team's detailed work on track without having to consider long-term strategy. Such strategies are left to management, which is exactly what the managers want.

On the other hand, this approach forces management to take early stands on technical issues related to the final product, since parts of the functionality are completed quite early in the project. Management sometimes found this decision process unsatisfactory.

In the ABB and Vodafone cases, the development was distributed over different locations. This clearly affected the chances of success. The greater the distance, the harder it was to coordinate the work to be completed and communicate how to do it.

About the Authors

Daniel Karlström is an associate researcher in the Software Engineering Research Group at Lund University, currently on sabbatical in Cambodia. His research interests address developer involvement in software process improvement, agile methods, maturity frameworks for software development, software quality, and project management. He received his PhD in software engineering from the Department of Communication Systems, Lund University. Contact him at the Dept. of Communication Systems, Lund Univ., Box 118, SE-221 00 Lund, Sweden; daniel.karlstrom@telecom.lth.se.

Per Runeson is a professor of software engineering at the Department of Communication Systems, Lund University, Sweden, and the leader of the Software Engineering Research Group. His research interests concern software development methods and processes, with focus on efficient and effective ways to facilitate software quality. He received his PhD in software engineering from Lund University. He's a senior member of the IEEE. Contact him at the Dept. of Communication Systems, Lund Univ., Box 118, SE-221 00 Lund, Sweden; per.runeson@telecom.lth.se.

Agile methods give the stage-gate model powerful tools for microplanning, day-to-day work control, and progress reporting. The functioning product and face-to-face meetings, for example, support much more powerful communications than written documents. The stage-gate model, in turn, gives agile methods a means to coordinate with other development teams and communicate with functions such as marketing and senior management.

On the basis of our observations, using XP teams in a stage-gate context is entirely feasible. We can suggest some adjustments both to engineering and project management that ease the integration.

The agile team must be prepared to interface with the gate model. This requires some overhead not usually associated with agile methods. However, as long as the need for this overhead is made clear and the overall effectiveness is demonstrated, our study indicates that this will work. In some instances, documentation can be eliminated; in others, it can be automatically generated. It should otherwise be treated as a development task. Including documentation in the time-boxed planning style makes it easy to prioritize the resources allocated to it.

While the technical staff generally appreciates agile methods, management sometimes fears them. Organizations must meet this management attitude up front for agile methods to serve them well.

If engineering does not actually initiate the change, they must at least be well anchored within the development teams. Universal acceptance is important for success. One way to achieve this is by making changes in small steps over a longer time.

All three studied cases showed the feasibility of integrating agile software development with stage-gate project management, thereby improving cost control, product functionality, and on-time delivery. The problems identified were primarily those that arise when introducing any new method. Although each of these projects involved just one XP team, we could not identify any issues indicating that integrating multiple XP teams into a stage-gate context would be more difficult.

References

1. K. Beck, "Embracing Change with Extreme Programming," *Computer*, Oct. 1999, pp. 70–77.
2. K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1999.
3. C. Wallin, F. Ekdahl, and S. Larsson, "Integrating Business and Software Development Models," *IEEE Software*, Nov./Dec. 2002, pp. 28–33.
4. M. Lindvall et al., "Agile Software Development in Large Organizations," *Computer*, Dec. 2004, pp. 26–34.