

# ICAM 2005 International Conference on Agility

## Helsinki July 27–28, 2005

### Life Cycle Model for Software Process Improvement Project Deploying an Agile Method

Tuomo Kähkönen

*Nokia Technology Platforms,  
P.O. Box 407, FIN-00045 Nokia Group, Finland*

**Abstract:** The focus of agile software development has been on methods and practices. How to take the methods effectively in use has received less attention. Especially in large organization it is not trivial to take agile methods in use. This paper presents two software process improvement projects from the initial idea of using agile approach to the end of the project. Based on the experiences, a life cycle model that describes phases in a software process improvement project deploying an agile method to an individual project was developed. The model defines the phases and the key decision points in the improvement project. It also defines some alternative processes for projects in different situations. The main benefit of the model is that it gives the big picture of the agile method deployment project. The model helps to understand which steps are needed for initially deploying an agile method and continuously improving the process.

## 1 Introduction

The focus in the agile software development research has been in the methods, i.e. which practices and principles to employ. How to effectively take the methods in use has received less attention. E.g. XP largely relies on experienced developers and a coach[1]. While this might be sufficient when the team is operating in agile home ground [2], this is not always enough. For example when large organizations are concerned, the team adopting an agile method encounters strong friction with its environment, e.g. the company wide mandatory processes. [3] Currently there is not available a systematic way to manage issues related to implementation of agile method in such situations.

As the main contribution, this paper presents a life cycle model for software process improvement (SPI) project that deploys an agile method in a single project. The author worked as a consultant in a team that helped business units in different types of SPI projects. A few years ago agile methods became one complementary approach among others [4]. It was acknowledged quite early that publicly available agile methods don't fit without modifications to teams operating in large organizations [3, 5]. Thus in every case an extensive tailoring was needed or the team had to develop its own agile approach (for examples, see [6]).

The agile methods provided very little implementation support. Coaching [1, 7] and post mortem workshops [8, 9] were used but that was not seen to be enough. Teams planning to take an agile method in use wanted to have a clear model how to take it in use and to tailor it to fit their specific needs.

The SPI team had used Software Engineering Institute's IDEAL<sup>1</sup> [10] as a reference model for conducting process improvement projects. Because the IDEAL model was originally a life cycle model for software process improvement based upon SW-CMM [11], it was perceived to be incompatible with agile approaches. It was thought that it is not feasible to have plan driven process improvement when the method itself was agile. Also the focus of the IDEAL was seen problematic because it focuses on organizations and the first initiatives in agile development had strong project focus. For these reasons there was a need for a light life cycle model for conducting SPI projects with projects using agile methods. Section two presents two case projects. The life cycle model for SPI project implementing an agile method is described in section three. Section four discusses the results of this study and finally concluding remarks are given.

## 2 Case Projects

This section describes how the two different case projects took agile methods in use, improved their processes and finally ramped down the process improvement activities. The focus of this section is on describing the deployment and tailoring activities, not the methods used. The author acted as a coach in both cases from the beginning of the adoption of agile approach until the end of the project. The case descriptions are based on coach's notes made during the projects and interviews carried out after the projects had finished. Although the model presented later in this paper embodies author's

---

<sup>1</sup> IDEAL is service mark of Carnegie Mellon University. CMM is registered in the US Patent and Trademark Office.

experiences from various agile and traditional projects during past seven years, these two cases provided a major contribution for the development of the model.

### 2.1 Case A - Established Team

Case A was an established team of 10 developers responsible for a single user interface intensive component of software platform. Although the organization had left considerable freedom for the team to select the way it worked, there were several processes that the team needed to conform. The team also had established their internal working practices and created considerable existing software asset. This project ended up with carefully planned incremental introduction of agile practices.

The improvement project was initiated with a presentation of agile approach to potential project managers, process manager and the line manager of the organization. As a result of the meeting, the line manager expressed his approval for the approach and there was a project manager that saw the proposed approach to be suitable for his team.

Before the start of the experiment the coach had developed a best practice model for agile development based on the process pattern idea [12]. The pattern model contained agile development practices identified from various agile development methods (e.g. [1, 13-15]) and some best practices identified within the company (e.g. [6, 16]). The best practice model was not intended to be as a complete methodology but rather a source of ideas for process developers.

The method building was started with short-listing of the patterns identified. Project manager selected with the coach and the process manager the patterns that were considered to be applicable for the team. They also identified some patterns that were already in use.

The team was informed about the forthcoming process improvement project in a kick-off meeting. The basis of agile software development was discussed, but the emphasis was on the motivation of the team members. To deepen the understanding about the agile practices available, the project manager and the process manager got familiar with a couple of key books on the agile development area.

The road from a pattern collection to a method was not straightforward. The implementation of the practices needed to be planned to follow a logical order. This was firstly due to the dependencies between the practices and secondly due to the limited capability of the team to take new practices in use. As all distortions to the team output needed to be avoided, only one or two practices could be changed at a time.

The process manager drew a dependency chart [17] of the practices for clarifying the roadmap (figure 1). The chart included existing practices and the planned ones. Arrows between the practices indicated perceived dependencies in the given context. The arrow width indicated the strength of the dependency. The chart was updated as new practices were taken in use.

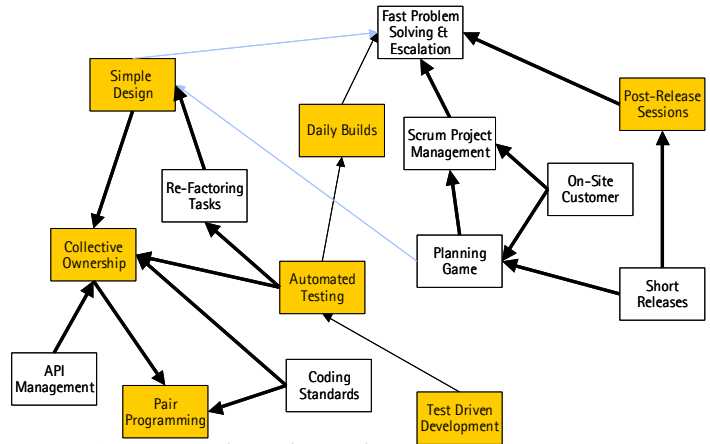


Figure 1: Practice dependency chart

The platform development organization used bi-weekly release cycles. The planning games were organized bi-weekly in synchronization with this cycle. In addition to this, the team had daily “Scrum style” [14] meetings and monthly post release sessions (based on the ideas at [8]). The planning game was for handling project content related issues and in the post release sessions the team had an opportunity to reflect which practices were working, which were having problems and how to improve the process. Daily meetings served for solving both project content and process related urgent issues.

In bi-weekly teleconferences project manager, process manager and coach discussed about the progress and the problems. The meeting helped to keep the process improvement activities in the project’s agenda also in the times when the business pressures were strong.

The first pilot lasted 10 months and the pilot was ended due to larger re-organization that discontinued the team. In case A many agile practices could be adopted without causing any major disturbance to business as usual. The crux of the implementation of further practices was automated unit testing that was not achieved. Although the product developed was ramped down, most team members continued together in a new team and continued using similar practices.

### 2.2 Case B - New Team

Case B was a new team of four that was established for developing a reporting system to be used in-house. This case ended up with a big-bang approach where most agile practices were taken in use in the first two iterations.

It was defined a priori that the team would use an agile method. The team defined the method with help from the coach. The main sources were XP [1] and MobileD [18]. Although the project was executed in a large organization, the team had only few process related pre-requisites. The project had to fulfill the milestone criteria set in the organization and the high level requirements had to be managed using a common requirement management process and system.

In the case B the coach had a dual role. He was both responsible for setting up the project and introducing the agile method in it. In addition to the agile development expertise, the coach had good domain knowledge because he had written the system previously used. The coach also knew the organization, its processes and relevant people.

In this project there was considerable amount of upfront work that enabled the team to start up quickly with the assignment. The team was set up in relatively short time compared with similar projects in the organization. The initiation included identification of stakeholders, finding budgeted and competent resources, acquiring facilities and equipment needed and establishing needed management processes. One important thing in setting up the project was to find a virtual customer who served as a proxy with different stakeholders and the team.

The project started development from scratches. The team didn't have any domain knowledge and no technical solution was imposed. The team started with selecting the programming language and development environment for the project.

The project started with one-day workshop where the coach and the virtual customer went through the system requirements and current technical environment. The goal of the workshop was to give an overview of the goals of the project and related domain and technical environment.

The project team then continued with a three-day spike (XP practice, see [1]) where the team set up the needed environment and implemented the simplest possible user story.

After the spike a two-day agile development training session was organized for the team. The method was laid out in a half-day workshop after the training. The coach made a proposal of applicable practices based on his experiences and the team decided after discussion which practices they apply and how.

In the first two-week iteration 19 of the selected 34 practices were adopted and in the second iteration most of the selected practices were at least partially in use. The adoption of the practices as well as their effectiveness was followed in post iteration workshops and the practices were tuned based on feedback. Metrics were collected throughout the project.

Process improvement actions were ended when the project went to maintenance mode. Only few essential agile practices and two original developers working part time were retained in the maintenance period.

### 2.3 Analysis of the Cases

When analyzed retrospectively, it is possible to identify from both cases four phases: initiation, ramp-up, continuous improvement and ramp down. It is prominent that the situation where the method deployment started was very different. The situation can be characterized with three overlapping life cycle models: team life cycle (e.g. [19]), project life cycle (e.g. [20-23]) and improvement project life cycle (e.g. [10]). While in

case A the team was already established and project work already ongoing, in case B all life cycles started simultaneously. This resulted that the coach had to help team also to establish foundations of successful project work and contribute to team building. The scope of the improvement activities was thus much wider than in the case of established team. This observation results a conclusion that we need at least two different processes for deploying an agile method.

Also the ramp-down of the improvement activities was different because case B went to low effort maintenance mode and case A was completely discontinued. This resulted that different activities were needed for benefiting from the accumulated process knowledge.

## 3 Generalized Model

Projects are in general divided to phases in order to provide better management control and appropriate links to the ongoing operations of the organization. Each phase ends up a review of project deliverables and performance in order to determine if the project should continue into its next phase and make possible corrective actions.[24] Based on the two cases it was possible to identify the following components of the improvement project:

- **Phases** that are typical for project deploying an agile method in a project or a team.
- **Milestones** that describe the main decision points in the deployment of agile method.
- **Processes** for deployment, continuous improvement and ramp-down. Different process variants were identified for different contexts and they are described using process goals and outputs.

The relationship of the components is described in the figure 2 and the milestones and phases are defined in the table 1.

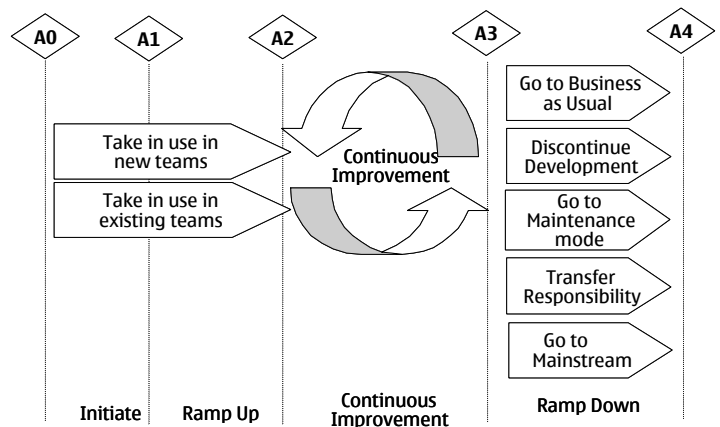


Figure 2: Improvement project life cycle model

The life cycle model has been designed so that the commonalities that based on the experiences are regarded to be applicable to all process improvement projects using agile

methods, are described in phases and milestones. The differences that originate from different situational factors are described in the different process variants.

Milestone/ Phase	Purpose
<b>A0</b>	Decision to investigate applicability of agile methods
<b>Initiate</b>	To select the right approach and create foundations for successful implementation
<b>A1</b>	Decision to take agile method in use
<b>Ramp-Up</b>	To involve development team and make initial deployment.
<b>A2</b>	Team has changed its operational model considerably. Agile improvement mechanisms in place.
<b>Continuous Improvement</b>	To refine methodology according to feedback and changing needs
<b>A3</b>	Decision to discontinue the process improvement activities
<b>Ramp Down</b>	To ensure the process knowledge produced by the project is used in the future to the maximum extent.
<b>A4</b>	Process improvement project finished

Table 1: Phases and milestones of improvement project

### 3.1 Initiate and Ramp-Up Phases

Initiate phase starts with an explicit decision to investigate applicability of agile methods. The initiate phase may be done in relatively small group including coach, senior manager, project manager and the key experts of the team. During the initiate phase it must be ensured that the project is in a situation where agile methods are a feasible option (for one analysis tool, see [25]). Also the environmental factors [3] should be considered – there might be external issues that prevent the use of agile approaches or impose important constraints. Also management commitment needs to be secured.

Initiate phase ends to a decision to take an agile method in use. It is followed by ramp-up phase where the whole team is involved. During the ramp-up phase some initial high-level planning is done in order to understand which process development steps are needed in long term. During the ramp-up phase the first improvement iteration is planned and executed. In the end of the phase the team has done first changes (or deployed the first practices if starting from scratch) and the mechanisms for continuous improvement are in place.

To conduct the first two phases of the model, two process variants were defined for the two basic scenarios of adopting agile methods.

#### 3.1.1 New Team Process

**Goal:** to set up an agile process for a new team quickly and to create a capability to improve it.

**Output:** a new team that is using an agile method and is capable to develop its own practices.

A small scout team can establish an agile team and its processes effectively. Three roles need to be present in the scout team in order to establish a productive team in shortest possible time:

- Sponsor, who is able to arrange needed resources, set up project steering structures and handle political issues related to the project.
- Domain expert, who is able to set project vision and goals and to define the key requirements for the project. He also helps the team to build up needed domain knowledge and to avoid common mistakes.
- Coach, who knows organization’s processes and practices as well as the agile methods and practices available. He helps the team to construct its processes and working practices so that the team complies to the organizational rules and within those limits operates as agile way as possible.

The scout team is responsible for the initiate phase in the new team process. In the beginning of the ramp-up phase the development team members receive needed training and agree their working practices with the help from the scout team. In practice this means that the first iteration is carried out with close support from the scout team.

#### 3.1.2 Established Team Process

**Goal:** to increase team’s ability to cope with changes without sacrificing business as usual.

**Output:** an existing team having more agile process and capability to improve it further.

In established team process the initiate and ramp-up phases are light when compared with the new team process. Initiate phase ensures that management commitment is in place and agile approach fits for the situation. The ramp-up phase serves as a transition from the initiate phase to continuous improvement phase - during the initiate phase the coach works mostly with the management, but in the continuous improvement phase the whole team is responsible for process development. In other words, the ramp-up phase involves the whole team, gets its commitment and trains it with the new improvement approach. Thus the first improvement iteration, where the team is learning by doing with the help from the coach, is a part of the ramp-up phase.

When an agile method is taken in use in an existing team, many traditional practices are already in use. During the ramp-up phase team needs to evaluate which practices can be dropped out or replaced by agile practices.

### 3.2 Continuous Improvement Phase and Process

**Goal:** to refine team’s existing process so that it better fits to business needs.

**Output:** new improved version of team’s process.

Process improvement in established agile teams has been studied quite much. Even the Agile Manifesto states that team should reflect how it is doing in regular intervals [26]. There are several papers which have studied how this reflection should be done (e.g. [8, 9]). Common approach for continuous improvement is a workshop where team analyzes what went well, what didn’t work and what could be improved. The workshop is called either post-mortem, post-release, post-iteration or reflection workshop (figure 3).

Metrics give another, quantitative, viewpoint to process improvement. There are some project management related metrics in the agile methods. However, the team can follow also more traditional product related metrics to get better understanding of the product quality. Based on the metrics, it is possible to evaluate whether the actions taken were effective or not. One example of light but sufficient metrics for agile projects can be found from [27]. Both case projects used adapted version of those metrics.

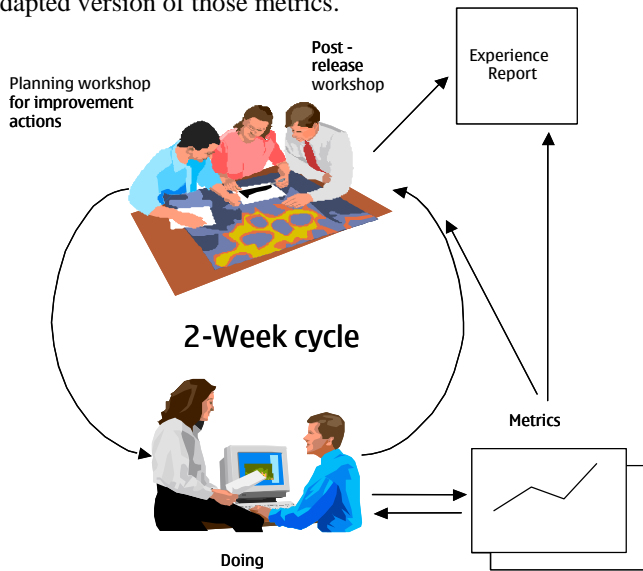


Figure 3: Continuous improvement cycle

### 3.3 Ramp Down Phase

The most evident reason for close down a process improvement project is that the team has refined its process to satisfactory level. However, experiences with the pilot cases have shown that volatility surrounding agile projects often results that improvement activities are finished for a business reason before the team has achieved a stable process where improvement opportunities are few. The reasons for

discontinuing improvement activities as well as the resulting state vary and thus also the activities needed in the ramping down the improvement project vary. For that reason the proposed model includes five different processes for ramp-down phase.

#### 3.3.1 Go to Business as Usual

**Goal:** to move from improvement project mode to continuous improvement mode.

**Output:** mechanisms in place that enable the team to maintain process discipline and identify further improvement opportunities as they rise.

If agile team has relatively long time span, it will eventually come to a point where the process has been stabilized and carrying on improvement in project mode does not deliver any added value. However, if the team just stop the improvement project, it faces a risk that the good practices deployed are forgotten. Thus the team must agree practices to make sure that the process discipline is maintained and further improvement opportunities are identified. Post iteration workshops are valid practice for this, although it may not be necessary to have them as frequently as before. Also the need for an external coach can be reconsidered.

#### 3.3.2 Discontinue Development

**Goal:** to utilize team’s experiences to maximum extent.

**Output:** accumulated process knowledge collected.

Sometimes it is realized that it is no use to continue the agile project for business reasons and there is no need for the actual results produced in the project. However, the project may have produced valuable lessons learned both regarding the development process and technologies used. The most effective option for keeping the accumulated process knowledge is to move the whole team to a new assignment [28]. If that is not feasible, it is possible to organize project post-mortem workshop [29, 30] for collecting and documenting experiences and lessons learned.

#### 3.3.3 Go to Mainstream

**Goal:** to change team’s process so that it meets the requirements set by the new business environment.

**Output:** team with more plan-driven process.

Agile approach fits very well to research projects and development of pilot versions. In some cases the experimental software developed is successful and it becomes a part of the mainstream development [31]. This may mean that the team size increases, team has more dependencies with other teams and thus the process needs to become more plan-driven.

From process improvement point of view this means that the process needs to be changed and the documentation needs to be developed to meet the new standards. It might be also necessary to reconsider the process improvement approach.

### 3.3.4 Transfer Responsibility

**Goal:** transfer development responsibility to another team.

**Output:** a new team capable to maintain and develop the software.

Sometimes the software developed by the agile project continues to live but the development is moved to another organization. Possible reasons for this kind of transfers are e.g. spin-off, selling the product to another company, moving to subcontracted development or restructuring of organizational responsibilities. The challenge in the transfer is that a great part of the product and process knowledge is tacit and thus it is difficult for the new team to assume the responsibility. One working solution for this dilemma has been apprenticeship period where the new team works with the original team [5]. Depending on the situation the scope of the knowledge transfer may be either the product related knowledge or include also the used development process.

### 3.3.5 Go to Maintenance Mode

**Goal:** ensure organization's capability to maintain the product effectively.

**Output:** maintenance strategy selected, needed processes agreed, resources and documents available.

In this concluding process for the improvement project the processes and practices needed during the maintenance period are agreed. There are different maintenance strategies that can be taken. Firstly, there is the XP way where the developers stay around [1, 32]. Secondly, the traditional approach where software is document to the extent where any software engineer can maintain it. There is also a middle-of-the-road approach where a few original developers will be available for maintenance at least for some time but also a minimal set of explanatory documents is written for mitigating the risk that the maintenance persons leave unexpectedly.

## 4 Discussion

The model presented in this paper was built based on experiences from actual projects. There are both similarities and differences to IDEAL model that represents traditional approach to SPI projects. Both models highlight continuous process improvement and there are similar phases and activities for planning, implementing and analyzing improvement actions. However, the way the activities are performed is different. It may be stated that when doing SPI

work with projects using agile methods, also the improvement approach should reflect the values of agile development [26, 33]. In practice this is visible in the following type of behavior (imitating the agile manifesto [26]):

- Putting emphasis on the people working in the software development team and involving them in the improvement initiative
- Delivering frequent effective changes to the process
- Collaboration with the customer and other stakeholders in order to find working practices that satisfy all
- Making small changes and reflecting often instead of making comprehensive improvement plans upfront

The most important individual difference between IDEAL and the proposed model is the time scale – while cycles in IDEAL model may last from months to up to a year, cycles in the proposed models last typically two weeks. Another important difference is the scope. While IDEAL model is designed for improving the capability of the whole organization, the model presented in this paper focuses on the needs of one project. As a consequence, the proposed model is lighter. Light but sufficient framework for SPI can be easily adopted also in agile projects and it still gives required structure for the improvement actions.

## 5 Conclusions

This paper presented a life cycle model for SPI project deploying an agile method in a single project. The model included project phases, milestones and process variants for different situations. The purpose of the model is to give the big picture of SPI project deploying an agile method.

The main finding of this paper to research community is that agile methods may benefit from process improvement methods especially designed for them. Waterfall approach with upfront planning and heavy documentation has influenced traditional SPI approaches. That mindset easily conflicts with agile methods. Thus more research on SPI methods that are iterative and puts emphasis on people is needed.

From practical point of view this paper offers one model that organizations can use for understanding better which steps are needed when an agile method is taken in use. The model also helps to make more explicit decisions during the SPI project and to have more structured yet evolutionary approach for implementing agile methods.

## Acknowledgements

I would like to thank Pertti Kontio and Veli-Matti Kananen from Nokia, Lasse Moisio and Timo Tenkanen from Affecto Oy and Harri Korpi and Seppo Sahi from Helsinki University of Technology for the participation to the data collection. I would like to thank Pekka Abrahamsson from VTT for giving his valuable comments on this paper.

## References

1. Beck, K., *Extreme Programming Explained: Embrace Change*. 2000, Upper Saddle River, NJ: Addison-Wesley. 190.
2. Boehm, B. and R. Turner, *Balancing Agility and Discipline*. 2003, Boston: Addison-Wesley.
3. Lindval, M., et al., *Agile Software Development in Large Organizations*. IEEE Computer, 2004(December 2004): p. 26-34.
4. Käsälä, K. *Good-Enough Software Process in Nokia*. in *Product Focused Software Process Improvement, Profess 2004*. 2004. Kansai Science City, Japan: Springer.
5. Vanhanen, J., J. Jartti, and T. Kahkonen. *Practical Experiences of Agility in the Telecom Industry*. in *XP 2003*. 2003.
6. Kähkönen, T. *Agile Methods for Large Organizations - Building Communities of Practice*. in *Agile Development Conference 2004*. 2003. Salt Lake City, Utah, USA.
7. Whitmore, J., *Coaching for Performance - Growing People, Performance and Purpose*. 1992, London: Nicholas Brealey Publishing.
8. Dingsøyr, T. and G.K. Hanssen. *Extending Agile Methods: Postmortem Reviews as Extended Feedback*. in *4th International Workshop on Learning Software Organizations*. 2002. Chicago, Illinois, USA.
9. Salo, O., et al. *Self-Adaptability of Agile Software Process: A Case Study on Post-Iteration Workshops*. in *5th International Conference on Extreme Programming and Agile Processes in Software Engineering*. 2004. Garmisch-Partenkirchen, Germany: Springer Verlag.
10. McFeeley, R., *IDEAL: A User's Guide for Software Process Improvement*. 1996, Pittsburgh, Pennsylvania: Carnegie Mellon University.
11. Gremba, J. and C. Myers, *The IDEAL Model: A Practical Guide for Improvement*. Bridge, 1997(3).
12. Coplien, J., *Software Patterns*. 1996, New York: SIGS Books & Multimedia.
13. Ambler, S.W. and R. Jeffries, *Agile Modeling*. 2002, New York: John Wiley & Sons Inc.
14. Schwaber, K. and M. Beedle, *Agile Software Development with Scrum*. Series in Agile Software Development, ed. R.C. Martin. 2002, Upper Saddle River: Prentice Hall.
15. Poppendieck, M. and T. Poppendieck, *Lean Software Development An Agile Toolkit*. The Agile Software Development Series, ed. A. Cockburn and J. Highsmith. 2003, Boston: Addison Wesley.
16. Kylmäkoski, R. *Efficient Authoring of Software Documentation Using RaPiD7*. in *25th International Conference on Software Engineering ICSE 2003*. 2003. Portland, Oregon.
17. Coplien, J.O. and N.B. Harrison, *Organizational Patterns of Agile Software Development*. 2004: Prentice Hall.
18. Abrahamsson, P., et al. *MobileD: An Agile Approach for Mobile Application Development*. in *OOPSLA 2004*. 2004. Canada.
19. Tuckman, B.W., *Developmental Sequence in Small Groups*. Psychological Bulletin, 1965. **63**(6): p. 384-399.
20. Royce, W. *Managing the Development of Large Software Systems*. in *IEEE Wescon*. 1970.
21. Boehm, B., *A spiral model of software development and enhancement*. Computer, 1988. **21**(5): p. 61-72.
22. Gilb, T., *Principles of Software Engineering Management*. 1988, Wokingham: Addison-Wesley.
23. *Rational Unified Process for System Engineering RUP SE 1.1*. 2002, Rational Software Corporation.
24. Project Management Institute, *Guide to the Project Management Body of Knowledge*. 2000: Project Management Institute.
25. Kettunen, P. and M. Laanti, *How to steer an embedded Software project: tactics for selecting the software process model*. Information and Software Technology, 2004. **In Press**.
26. Beck, K., et al., *Manifesto for Agile Software Development*. <http://www.agilemanifesto.org>. 2001.
27. Abrahamsson, P. *Extreme Programming: First Results from a Controlled Case Study*. in *29th Euromicro Conference*. 2003. Belek, Antalya, Turkey.
28. DeMarco, T. and T. Lister, *Peopleware: Productive Projects and Teams, 2nd Edition*. 1999, New York: Dorset House Publishing.
29. Tiedeman, M., *Post-Mortems - Methodology and Experiences*. IEEE Journal on Selected Areas in Communications, 1990. **8**(2): p. 176-180.
30. Collier, B., T. DeMarco, and P. Fearey, *A Defined Process For Project Postmortem Review*. IEEE Software, 1996. **13**: p. 65-72.
31. Moore, G.A., *Crossing the Chasm*. 1991: Harper Business.
32. Poole, C.J., et al. *Extreme Maintenance*. in *IEEE International Conference on Software Maintenance, 2001*. 2001. Florence, Italy.
33. Fowler, M. and J. Highsmith, *The Agile Manifesto*. Software Development Magazine, 2001(August 2001).