

On Empirical Research Into Scrum

Dr. Mark C. Paulk
Institute for Software Research
Carnegie Mellon University
5000 Forbes Avenue, 407SCR 115
Pittsburgh, PA 15213
Email: mcp@cs.cmu.edu

Noopur Davis
Davis Systems
Pittsburgh, PA
Email: ndavis@davissys.com

Abstract

The agile methods, such as Scrum and Extreme Programming (XP), have been a topic of much discussion in the software community over the last few years. While the proponents of the agile methods have articulated convincing arguments for their methods, usually within a context of small-to-medium size projects with significant requirements volatility, opponents have expressed serious concerns about the appropriateness and effectiveness of the methods. The research project described in this report is three-pronged effort to investigate the issues associated with Scrum adoption. First, the practices that characterize the Scrum agile method will be stated, along with common variants. Second, projects that have adopted, or are in the process of adopting, Scrum will be surveyed to identify which Scrum practices, or variants thereof, they have implemented and the perceived value of the method. Third, factors affecting Scrum adoption will be investigated. The objective of this research is to better understand the barriers to adoption and the leverage points that might encourage Scrum to be more widely and efficiently deployed.

Introduction	3
Engineering and Management Practices	5
Scrum Practices	5
The Product Backlog	6
The Sprint	6
The Daily Scrum	7
The Scrum Roles	7
Potential Survey Questions Related to Scrum Practices	8
Other Engineering Practices	9
Potential Survey Questions Related to Other Engineering Practices	10
Defining Success	12
Project Success Dimensions	12
Balanced Scorecard at the Organizational Level	12
The Discipline of Market Leaders	12
Potential Survey Questions Related to Defining Success	12
Change Management and Scrum Adoption	13
Diffusion of Innovations	13
Crossing the Chasm	13
The Assimilation Gap	13
The Innovation Analysis Model	13

Potential Survey Questions Related to Change Management	13
Improvement Factors	14
TQM Factors	14
“Buy-In” to Improvement.....	14
Critical Success Factors for Improvement	14
Readiness to Change.....	14
Why Organizations Do Not Change.....	14
Evidence-Based Management	14
Potential Survey Questions Related to Improvement Factors	14
Cultural Factors Affecting Change Management	15
Hofstede’s Software of the Mind	15
Constantine’s Organizational Paradigms	15
Handy’s Gods of Management	15
Schein’s Organizational Cultures	15
Potential Survey Questions Related to Culture	15
Survey Design.....	16
Sampling and Bias Concerns.....	16
An Adequate Response Rate	17
Next Steps and Future Research	18
References	19

INTRODUCTION

The agile methods, such as Scrum and Extreme Programming (XP), have been a topic of much discussion in the software community in recent years. While the proponents of the agile methods have articulated convincing arguments for their methods, usually within a context of small-to-medium size projects with significant requirements volatility, opponents have expressed serious concerns about the appropriateness and effectiveness of the methods. This report describes a three-pronged empirical research project into Scrum adoption. First, the practices that characterize the Scrum agile method will be stated, along with common variants and tailorings. To understand Scrum adoption, we need to clearly identify what is an acceptable implementation of Scrum and what is not. Second, projects that have adopted, or are in the process of adopting, Scrum will be surveyed to identify which Scrum practices, or variants thereof, they have implemented and the perceived value of the method. Third, factors affecting Scrum adoption will be investigated.

Authorities from the Scrum Alliance, e.g., Schwaber and Sutherland, will be consulted to identify what is, and is not, a reasonable tailoring of Scrum. The interesting questions from an empirical research perspective include:

- What are the critical Scrum practices to consider?
- What variations on each of the Scrum practices occur in projects?
- What tailorings are legitimate variations that a project can use and be considered as following the Scrum method?

Industry projects that have adopted, or are in the process of adopting, Scrum will be surveyed to identify which Scrum practices, or variants thereof, they have implemented and the perceived value of the method/practices. For those choosing whether to adopt Scrum, understanding the benefits is crucial to the decision. Formal cost/benefit analyses, however, are rarely performed by software organizations (high maturity organizations as rated against the Software CMM or CMM Integration are likely to be exceptions). While objective measures of cost and benefit would be preferred, and will be collected to the degree available, rigorous measures of value are frequently lacking in industry projects, so this first round of research will be based on the perception of value.

Industry projects that have adopted, or are in the process of adopting, Scrum will be surveyed to identify how they define success and how well Scrum supports projects in being successful. A prerequisite to investigating value is understanding what success means in a specific context. In some cases, success is driven by cost and schedule predictability; for example, in government contracting, predictability and operational excellence are highly valued. In some cases, success is based on functionality delivered and the relationship of that functionality to business objectives. Building a mutually beneficial relationship with the customer can also be considered a measure of success. Understanding the business context will clarify how Scrum supports achieving successful projects in that environment. The interesting questions from an empirical research perspective include:

- How is success determined?
- What practices are perceived to work well or badly on Scrum projects?
- What measures of success do Scrum projects use (if any) for the method itself?

Industry projects that have adopted, or are in the process of adopting, Scrum will be surveyed to identify what factors have influenced the adoption of Scrum. Many sources can be used to identify potential factors, including those affecting the diffusion of innovations, those affecting the marketing of new technologies and products, and those factors that influence the success of software process improvement efforts. Although there is great overlap in the concerns of these different research areas, each adds a unique perspective that may enlighten the research. The interesting questions from an empirical research perspective include:

- What external factors, e.g., access to user groups, affect Scrum adoption?
- What internal factors, e.g., training, affect Scrum adoption?

In this report we provide a brief overview of each of the topics we wish to investigate in this research, followed by an identification of the specific items that we might wish to investigate in the context of Scrum adoption. The overviews identify important attributes of the topics but are not comprehensive descriptions; similarly the specific items to investigate filter out many important attributes to focus on particularly relevant information. The surveys that will be distributed provide another round of filtering (survey design and usability issues are described in a later section of this report). Each round of filtering is intended to help us focus on the vital few issues that materially affect Scrum adoption of the hundreds of possibilities. We anticipate that this initial round of survey-based research will identify a number of questions worthy of further investigation. We hope that some of the companies involved will consider participating in a more rigorous set of case studies, but that investigation is not considered in this report.

ENGINEERING AND MANAGEMENT PRACTICES

Stating the practices that characterize the Scrum agile method is challenging because Scrum is not a software engineering or development methodology in a strict sense, although it could be described as a project management methodology. It is more of a philosophy or set of values that defines a culture of empowerment and participation within the team and of collaboration and transparency with the customer. Still, it can be characterized by a relatively small set of practices and roles originally established by Ken Schwaber and Jeff Sutherland, but these practices must be interpreted in light of agile values and principles.

Elaborating this point, Ken Schwaber commented:*

Scrum is a tool, a framework, that can be used to build complex products. It does not prescribe any of the common engineering, people, risk management, or other practices. For instance, it doesn't say the team has to be collocated.

What Scrum does provide is feedback so that someone using Scrum can improve the results. For instance, if someone wants productivity and quality and can have a collocated team, Scrum will point this out. If the person starts with a dispersed team and compares its productivity to another collocated team, conclusions can be reached. An intelligent person would then change (continuous process improvement).

So using Scrum correctly means following all of its rules, which expose everything (transparently) for inspection and adaptation.

An intelligent person would then inspect what Scrum is making transparent and make changes to optimize the results. Presumably, the changes are cost justified.

Someone can use Scrum perfectly and ignore what is made transparent.

Someone can use Scrum imperfectly and act on some of the things that have been made transparent.

Someone who uses Scrum perfectly and acts more intelligently than anyone else on what has been made transparent will outcompete anyone else.

Since Scrum does not explicitly address engineering practices, it is desirable to consider non-Scrum practices that may be tightly linked to Scrum success. For example, test-driven development is frequently advocated for agile projects but is not an explicit Scrum practice. Other methods can act as a source of potential practices that may influence the success of Scrum, because Scrum is frequently implemented in conjunction with methods such as Extreme Programming.

Scrum Practices

Schwaber's two books [Schwaber02, Schwaber04] can be considered the definitive statements of what Scrum is, although the Scrum Alliance has recently published a "Scrum Guide" that can be considered the formal definition of the method [SA09]. Vodde and Sutherland created the Nokia Test to assess the status of teams claiming to use Scrum [Sutherland08]. Silver, another Scrum

* Email to Mark Paulk, "Re: a start on identifying survey topics," 27 April 2009.

Alliance member, has also identified crucial characteristics and practices for Scrum [Silver07]. These descriptions are elaborated and clarified in various reports and training, as well as related books [Cohn05, Larman08].

Scrum can be summarized as three practices and three roles. The Product Backlog, Sprint, and Daily Scrum are the three primary practices, although there are a number of related practices. The ScrumMaster, Product Owner, and Development Team are the three essential roles. The Scrum Team consists of people performing these three roles. The following brief description from the “Scrum Guide” highlights the specific aspects of Scrum that we may wish to investigate to verify that the Scrum implementation is a valid one.

Scrum projects are typically small to medium sized, but large Scrum projects have been reported. These are typically organized as a “Scrum of Scrums” [Schwaber04]. Scrum has also been adopted at the enterprise level [Schwaber07]. Agile methods in general are assumed to be geographically co-located, but distributed (virtual) teams have been described [Ramesh06], although large and distributed projects are quite different from the environment assumed for agile methods in general. High requirements volatility is usually assumed for agile projects. Cockburn characterizes the sweet spots for agile projects as two to eight people in one room, onsite usage experts, one-month increments, fully automated regression tests, and experienced developers [Cockburn02].

In many cases Scrum is adopted as a whole with little change, but in some cases it is adopted in a “tailored” form. This tailoring may, or may not, represent a reasonable adaptation of the original method. Inappropriate Scrum variations are colloquially known as “ScrumButs.” Thus the perceived need for the Nokia Test. It is therefore necessary to investigate the implementation to determine whether a failed Scrum implementation truly reflects the method or an inappropriate and ineffective understanding of what Scrum is. It seems likely that Scrum is a “bundle” of knowledge that is best adopted as a whole [MacDuffie95, Pil96]; piecemeal adoption of Scrum practices is unlikely to achieve the emergent behaviors and benefits of the method.

The Product Backlog

The Product Backlog lists the requirements for the product being developed. It is the master list of all functionality desired in the product, and each item in the Product Backlog has a description, a priority and an estimate of the effort needed to complete it.

The Release Plan describes the goal of the release, the highest priority items in the Product Backlog, the major risks, and the overall features and functionality that the release will contain. It establishes a probable delivery date and cost, assuming that nothing changes.

The Sprint

A Sprint is one iteration of a month or less that is of consistent length throughout a development effort. Only the Product Owner has the authority to cancel the Sprint. Vodde and Sutherland suggest that Sprints may be 2-6 weeks long.

The Sprint Planning Meeting is when the iteration is planned. It is time boxed to eight hours (for a one month Sprint) and has two parts: determining what will be done in the Sprint and how the Team is going to build the product increment during the Sprint.

The Sprint Backlog is an output of the Sprint Planning Meeting. It consists of the tasks, task estimates, and assignments for the Sprint. Only the Team can change the items or their estimates during a Sprint.

“Done” defines what the Team means when they commit to “doing” a Product Backlog item in a Sprint. A completely “done” increment includes all of the analysis, design, refactoring, programming, documentation and testing for the increment and all Product Backlog items in the increment.

The Sprint Backlog Burndown is a graph of the amount of Sprint Backlog work remaining in a Sprint across time in the Sprint. The Release Burndown graph records the sum of remaining Product Backlog estimated effort across time. The velocity (or productivity) of the team can be calculated using the burndown charts.

The Sprint Review meeting is a four-hour time-boxed meeting (for one-month Sprints) that is held at the end of a Sprint where the Team presents the functionality done in the iteration to the Product Owner and other stakeholders. The Team demonstrates the work that is done and answers questions. The Team discusses what went well during the Sprint and what problems they ran into, and how they solved these problems.

The Sprint Retrospective meeting is a three hour, time-boxed meeting (for one-month Sprints) held after the Sprint Review and prior to the next Sprint Planning meeting where the Team discusses what went well in the last Sprint and what can be improved for the next Sprint.

The Daily Scrum

The Daily Scrum is a 15-minute meeting used to inspect progress toward the Sprint goal and to make adaptations that optimize the value of the next workday. During the meeting, each Team member explains:

- 1) What he or she has done since the last Daily Scrum.
- 2) What he or she is going to do before the next Daily Scrum.
- 3) What obstacles are in his or her way.

The Scrum Roles

The ScrumMaster is the specific individual responsible for ensuring that Scrum values, practices and rules are enacted and enforced. The ScrumMaster is the project manager but leads by coaching, teaching and supporting the Team rather than directing and controlling.

The Product Owner is the specific individual responsible for managing and controlling the Product Backlog. The Product Owner sets the priority for each item in the Product Backlog.

The Team is typically seven people, plus or minus two. Teams are cross-functional, having all the skills needed to create an increment.

Potential Survey Questions Related to Scrum Practices

Is there a single ScrumMaster? Is he or she a Certified ScrumMaster? Is there a project manager separate from the ScrumMaster?

Is there a single Product Owner? Does the Product Owner integrate and reconcile the desires of multiple stakeholders? Is the Product Owner co-located with the Development Team?

How many people are on the Development Team? Is the Development Team co-located? Distributed across multiple sites? What is the average experience (in years) of the team in building software? The minimum? The maximum?

Is Scrum being piloted on this project? Has Scrum been deployed across the organization (made available to the entire company or the business unit)? Is senior management sponsoring the adoption of Scrum?

How many months has the Scrum project been running?

What kind of application is the Scrum project building (e.g., Web, MIS, embedded system)?

Is the Scrum project a single project or part of a larger program, e.g., a Scrum of Scrums?

Does the Product Backlog consist of user stories with associated priorities and estimates? Is there a requirements specification maintained under formal change control?

Are Sprints consistently 30 days long? Less than 30 days? More than 30 days but less than six weeks? Variable length between two and six weeks? More than six weeks?

Are the Daily Scrum meetings held every day? Held multiple times per week but not necessarily daily? If part of a Scrum of Scrums, are Daily Scrum meetings held at the higher levels in the hierarchy?

Is there a common understanding of what “done” means for the items in the Sprint Backlog? Are there regression tests to demonstrate that functional requirements have been correctly implemented that are performed on a daily or more frequent basis?

Is the Release Plan based on the known velocity of the project and the desired functionality at the time of the release?

Is a Sprint Retrospective Meeting held at the end of Sprints to identify opportunities for improvement?

Is the Scrum project viewed by the various stakeholders as being open and transparent? Is the Scrum Team viewed as being collaborative and responsive? Is the Scrum Team viewed as being competent, knowledgeable, and professional?

Other Engineering Practices

Probably the two most popular agile methods are Scrum and Extreme Programming (XP). Beck described in terms of twelve practices in the first edition of his book *Extreme Programming Explained: Embrace Change* [Beck99]:

- Planning game
- Small releases
- Metaphor
- Simple design
- Testing (including test-driven development and customer tests)
- Refactoring (described as design improvement by some)
- Pair programming
- Collective (code) ownership
- Continuous integration
- 40-hour week (later described as sustainable pace)
- On-site customer
- Coding standard

In the second edition [Beck04], XP is described in terms of primary practices:

- Sit together
- Whole team
- Informative workspace
- Energized work
- Pair programming
- Stories
- Weekly cycle
- Quarterly cycle
- Slack
- Ten-minute build
- Continuous integration
- Test-first programming
- Incremental design

and corollary practices:

- Real customer involvement
- Incremental deployment
- Team continuity
- Shrinking teams
- Root-cause analysis
- Shared code
- Code and tests
- Single code base
- Daily deployment
- Negotiated scope contract

- Pay-per-use

XP and Scrum are a popular hybrid agile method. XP practices that may be particularly pertinent to successful Scrum adoption include test-driven development, refactoring, pair programming, and sustainable pace. Stephens and Rosenberg discuss a number of concerns about XP in their book *Extreme Programming Refactored* [Stephens03]. Among their concerns are the potential for continuous integration to become occasional integration, the customer not to be available as needed, and teams to not have the generalist skills and competence necessary to do the work. These concerns are valid for Scrum (and other agile methods) as well.

Risk management is fundamental to successful software project management [Boehm91, Charette96]. Boehm identifies a set of common software project risks that can act as a starting point:

- Personnel shortfalls
- Unrealistic schedules and budgets
- Developing the wrong functions and properties
- Developing the wrong user interface.
- Gold-plating.
- Continuing stream of requirements changes.
- Shortfalls in externally furnished components.
- Shortfalls in externally performed tasks.
- Real-time performance shortfalls.
- Straining computer science capabilities.

Many of the practices in the agile methods are mechanisms for managing risks effectively, e.g., one-month iterations help manage the risk of requirements volatility. While many of these risks, such as a continuing stream of requirements changes, are intrinsically addressed in the agile methods, some, such as shortfalls in externally furnished components or externally performed tasks, can still be concerns. A potential good management practice is identifying and monitoring a set of “top-10” risks.

Similarly, concurrent engineering (also known as integrated process and product development) [Blackburn96, Smith97], is another well-known technique that the agile methods typically address. Cross-functional teams, a focus on the customer, and the use of lead time as a source of competitive advantage are intrinsic to the agile methods, therefore agile methods can be considered a form of concurrent engineering with the caveat that all the needed skills are represented on the Development Team.

Potential Survey Questions Related to Other Engineering Practices

Does the Development Team do test-driven development?

Does the Development Team do pair programming?

Does the Development Team do simple designs, with refactoring as appropriate?

Does the Development Team work at a sustainable pace?

Are risks identified and monitored at the end of each Sprint?

Does the Development Team have all the skills needed to do the work?

NEXT STEPS AND FUTURE RESEARCH

This report describes empirical research into Scrum implementation and adoption that is planned. The next steps are to design the survey instrument, have it reviewed by selected colleagues from academia and members of the Scrum Alliance, administer the survey to the three samples, analyze the data, and publish the results. Those results may inform subsequent actions by various stakeholders in encouraging and enabling Scrum adoption.

The research begins with a Web-based survey, but follow-up in the form of e-mail, telephone interviews, and on-site discussions is likely to be necessary to clarify the data and expand our insights. These follow-ups may result in case studies of Scrum adoption by specific organizations, but those case studies are potential future research that is outside the scope of this report.

We may reasonably expect this research will spark further questions that may be addressed via additional surveys. Those surveys are potential future research that is outside the scope of this report.

This research project will focus on industry projects, but the studio projects for Carnegie Mellon's Masters in Software Engineering (MSE) program offer the opportunity for focused case studies. Whether the insights available from a student context are worth actively pursuing will be discussed with the various stakeholders. Any research conducted in the MSE environment are potential future research that is outside the scope of this report..

REFERENCES

- Beck99 K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, Boston, 1999.
- Beck04 K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change, 2nd Edition*, Addison Wesley, Boston, 2004.
- Blackburn96 J.D. Blackburn, G. Hoedemaker, and L.N. Van Wassenhove, "Concurrent Software Engineering: Prospects and Pitfalls," *IEEE Transactions on Engineering Management*, Vol. 43, No. 2, May 1996, pp. 179-188.
- Boehm91 B.W. Boehm, "Software Risk Management: Principles and Practices," *IEEE Software*, Vol. 8, No. 1, January 1991, pp. 32-41.
- Boehm04 B.W. Boehm and R. Turner, *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison-Wesley, Boston, 2004.
- Charette96 R.N. Charette, "Large-Scale Project Management is Risk Management," *IEEE Software*, Vol. 13, No. 4, July 1996, pp. 110-117.
- Chrissis06 M.B. Chrissis, M.D. Konrad, and S. Shrum, *CMMI: Guidelines for Process Integration and Product Improvement, Second Edition*, Addison-Wesley, Boston, 2006.
- Cockburn02 A. Cockburn, *Agile Software Development*, Addison-Wesley, Boston, 2002.
- Cohn05 M. Cohn, *Agile Estimating and Planning*, Prentice Hall, 2005.
- Constantine95 L.L. Constantine, *Constantine on Peopleware*, Yourdon Press, Englewood Cliffs, NJ, 1995.
- Daghfous91 A. Dagfous and G.R. White, "Information and Innovation: A Comprehensive Representation," University of Pittsburgh, Department of Industrial Engineering, Technical Report 91-4, 1991.
- Fenton94 N.E. Fenton, S.L. Pfleeger, and R.L. Glass, "Science and Substance: A Challenge to Software Engineers," *IEEE Software*, Vol. 11, No. 4, July 1994, pp. 86-95.
- Fichman99 R.G. Fichman and C.F. Kemerer, "The Illusory Diffusion of Innovations: An Examination of Assimilation Gaps," *Information Systems Research*, Vol. 10, No. 3, September 1999, pp. 255-275.
- Gladwell05 M. Gladwell, *Blink - The Power of Thinking Without Thinking*, Little, Brown, and Company, New York, 2005.
- Goldratt97 E.M. Goldratt, *Critical Chain*, North River Press, Great Barrington, MA, 1997.
- Handy91 C. Handy, *Gods of Management: The Changing Work of Organizations, Third Edition*, Oxford University Press, New York, 1991.
- Hofstede96 G. Hofstede, *Cultures and Organizations, Software of the Mind: Intercultural Cooperation and its Importance for Survival*, McGraw-Hill, New York, 1996.

- ISO 9126-1 “Software Engineering - Product Quality - Part 1: Quality Model,” International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 9126-1, 2001.
- Kaplan96 R.S. Kaplan and D.P. Norton, *The Balanced Scorecard: Translating Strategy into Action*, Harvard Business School Publishing, Boston, 1996.
- Larman08 C. Larman and B. Vodde, *Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum*, Addison-Wesley, Boston, 2008.
- LeGault 06 M.R. LeGault, *Think! Why Crucial Decisions Can't Be Made in the Blink of an Eye*, Threshold Editions, 2006.
- MacDuffie95 J.P. MacDuffie, “Human Resource Bundles and Manufacturing Performance: Organizational Logic and Flexible Production Systems in the World Auto Industry,” *Industrial and Labor Relations Review*, Vol. 48, No. 2, January 1995, pp. 197-221.
- McGarry02 J. McGarry, D.N. Card, C. Jones, B. Layman, E. Clark, J. Dean, and F. Hall, *Practical Software Measurement: Objective Information for Decision Makers*, Addison-Wesley, Boston, 2002.
- Moore91 G.A. Moore, *Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers*, HarperCollins, New York, 1991.
- Niazi06 M. Niazi, D. Wilson, and D. Zowghi, “Critical Success Factors for Software Process Improvement Implementation: An Empirical Study,” *Software Process Improvement and Practice*, Vol. 11, No. 2, March/April 2006, pp. 193-211.
- Paulk95 M.C. Paulk, C.V. Weber, B. Curtis, and M.B. Chrissis, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley, Boston, 1995.
- Paulk99 M.C. Paulk, “Structured Approaches to Managing Change,” *Crosstalk: the Journal of Defense Software Engineering*, Vol. 12, No. 11, November 1999, pp. 4-7.
- Paulk02 M.C. Paulk, “Agile Methodologies and Process Discipline,” *Crosstalk: the Journal of Defense Software Engineering*, Vol. 15, No. 10, October 2002, pp. 15-18.
- Pfeffer06 J. Pfeffer and R.I. Sutton, *Hard Facts, Dangerous Half-Truths, & Total Nonsense: Profiting from Evidence-Based Management*, Harvard Business School Press, Boston, 2006.
- Pil96 F.K. Pil and J.P. MacDuffie, “The Adoption of High-Involvement Work Practices,” *Industrial Relations*, Vol. 35, No. 3, July 1996, pp. 423-455.
- Pinto02 J.K. Pinto, “Project Management 2002,” *Research Technology Management*, March/April 2002, pp. 22-37.
- Rainer03 A. Rainer and T. Hall, “A Quantitative and Qualitative Analysis of Factors Affecting Software Processes,” *The Journal of Systems and Software*, Vol. 66, No. 1, April 2003, pp. 7-21.

- Rainer03a A. Rainer, T. Hall, and N. Baddoo, "Persuading Developers to 'Buy Into' Software Process Improvement: Local Opinion and Empirical Evidence," Proceedings of the 2003 International Symposium on Empirical Software Engineering, 2003.
- Ramesh06 B. Ramesh, L. Cao, K. Mohan, and P. Xu, "Can Distributed Software Development Be Agile?" Communications of the ACM, Vol. 49, No. 10, October 2006, pp. 41-46.
- Rea05 L.M. Rea and R.A. Parker, *Designing and Conducting Survey Research: A Comprehensive Guide, Third Edition*, Jossey-Bass, San Francisco, 2005.
- Reifer03 D.J. Reifer, "Is the Software Engineering State of the Practice Getting Closer to the State of the Art?" IEEE Software, Vol. 20, No. 6, November/December 2003, pp. 78-83.
- Rogers03 E.M. Rogers, *Diffusion of Innovations, Fifth Edition*, The Free Press, New York, 2003.
- Rousseau07 D.M. Rousseau and S. McCarthy, "Educating Managers From an Evidence-Based Perspective," Academy of Management Learning & Education, Vol. 6, No. 1, March 2007, pp. 84-101.
- SA09 "Scrum Guide," Scrum Alliance, <URL: <http://www.scrumalliance.org/resources>>, 2009.
- Schein92 E.H. Schein, *Organizational Culture and Leadership, Second Edition*, Jossey-Bass, San Francisco, 1992.
- Schwaber02 K. Schwaber and M. Beedle, *Agile Software Development with Scrum*, Prentice-Hall, Upper Saddle River, NJ, 2002.
- Schwaber04 K. Schwaber, *Agile Project Management with Scrum*, Microsoft Press, Redmond, WA, 2004.
- Schwaber07 K. Schwaber, *The Enterprise and Scrum*, Microsoft Press, Redmond, WA, 2007.
- Shenhar97 A.J. Shenhar, O. Levy, and D. Dvir, "Mapping the Dimensions of Project Success," Project Management Journal, Vol. 28, No. 2, June 1997, pp. 5-13.
- Silver07 M.G. Silver, "Am I, or Am I Not, Using Scrum? That Is the Question," Scrum Alliance, 18 March 2007.
- Smith97 R.P. Smith, "The Historical Roots of Concurrent Engineering Fundamentals," IEEE Transactions on Engineering Management, Vol. 44, No. 1, February 1997, pp. 67-78.
- Staples07 M. Staples, M. Niazi, R. Jeffery, A. Abrahams, P. Byatt, and R. Murphy, "An Exploratory Study of Why Organizations Do Not Adopt CMMI," The Journal of Systems and Software, Vol. 80, No. 6, June 2007, pp. 883-895.
- Stephens03 M. Stephens and D. Rosenberg, *Extreme Programming Refactored: The Case Against XP*, Apress, Berkeley, 2003.

- Sutherland08 J. Sutherland and B. Vodde, "The Nokia Test, Extended with Scoring by Jeff Sutherland," <URL: http://www.cedur.se/nokia_test2.html>.
- Thayer97 R.H. Thayer and R.E. Fairley, "Software Engineering Project Management: The Silver Bullets of Software Engineering," in *Software Engineering Project Management, Second Edition*, R.H. Thayer (ed), 1997, pp. 504-505.
- Treacy97 M. Treacy and F. Wiersema, *The Discipline of Market Leaders*, Addison-Wesley, Boston, 1997.