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

Larry Maccherone
Institute for Software Research
Carnegie Mellon University

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.

Keywords: agile methods, change management, innovation, methodology, Scrum, software process, technology adoption, Total Quality Management, TQM

Acknowledgements: Thanks to the Scrum Alliance for their support of this research. A number of people have contributed comments on various drafts of this report, including Karthik Dinakar, Scott Duncan, Ken Schwaber, Jeff Sutherland, and Bas Vodde.

Table of Contents

Introduction 3
Engineering and Management Practices 5
  Scrum Practices and Roles 6
  Other Engineering Practices 8
  The Agile Culture 9
Defining Success 11
Change Management and Scrum Adoption 14
Improvement Factors 18
Cultural Factors Affecting Change Management 20
Survey Design 21
  Sampling and Bias Concerns 21
  An Adequate Response Rate 22
Next Steps and Future Research 23
References 24
Appendix A. Questions for a Comprehensive Survey 30
  Respondent Demographics 30
  Project Demographics 30
  Organizational Demographics 30
  Scrum Practices 30
  Other Engineering Practices 31
  Agile Culture 32
  Defining Success 32
  Change Management 32
  Process Improvement Factors 33
  Cultural Factors 33
Appendix B. The Scrum Adoption Survey 34
  The Invitation 34
  The Survey 34

INTRODUCTION

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. Before we can investigate the factors associated with Scrum adoption, we must be sure that we are considering a legitimate Scrum implementation. Some interesting questions about the Scrum method itself 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.

Organizations 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. Some of the interesting questions about what success means for a Scrum project include:

- How is success determined for the project?
- 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?
- How successful are Scrum projects in terms of their measures of success?

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. Some of the interesting questions about factors affecting Scrum adoption 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.

This research will not address criticisms of the agile methods. 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. While Extreme Programming has been the focus of many of these concerns [Keefer03, Stephens03], general criticisms of agile methods include: agile methods may not accommodate the working style of the organization's best programmers [Skowronski04]; there may be a culture clash between the customer's way of doing things and the agile approach [Paulk02]; agile projects require "premium" or "superbly trained" people [DeMarco02]; and agile methods legitimize hacker behavior [Rakitin01]. Boehm suggests that "both agile and plan-driven approaches have a responsible center and over-interpreting radical fringes" [Boehm02]. A thoughtful and informed choice of what methods are appropriate for a given project context is needed [Boehm04, Larman04]. This research will investigate some of the factors associated with the success (or failure) of Scrum projects, which may touch on some of these issues, but the focus of the research is not to rebut such criticisms.

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. Some of the practices below are not universally considered part of Scrum and are noted as appropriate.

Elaborating this point, Schwaber[1] 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 co-located.
>
> 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 co-located team, Scrum will point this out. If the person starts with a dispersed team and compares its productivity to another co-located 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 out-compete 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.

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].

[1] K. Schwaber, email to Mark Paulk entitled “Re: a start on identifying survey topics,” 27 April 2009.

Scrum Practices and Roles

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 may be considered the formal definition of the method [SA09]. Sutherland and Vodde created the Nokia Test to assess the status of teams claiming to use Scrum [Sutherland08]. Silver, another Scrum Alliance member, has also identified crucial characteristics and practices for Scrum [Silver07]. These descriptions of Scrum and its practices are elaborated and clarified in various reports and training, as well as related books [Cohn05, Larman08]. The following brief description of Scrum practices and roles highlights the specific aspects of Scrum that we may wish to investigate to verify that a Scrum implementation is a valid one.

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 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. In reviewing a draft of this report, Vodde commented that a Release Plan is not part of Scrum. A Release Plan is described in several Scrum descriptions [SA09, Cohn05], including the Nokia Test written by Sutherland and Vodde [Sutherland08], but not all [Schwaber02, Schwaber04, Silver07].

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. Sutherland and Vodde 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 of the tasks for the Sprint derived from the Product Backlog. "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 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 and discusses the work done in the Sprint.

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 is a time-boxed, 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 ScrumMaster is the specific individual responsible for ensuring that Scrum values, practices and rules are enacted and enforced. Some would characterize the ScrumMaster as the project manager who leads by coaching, teaching and supporting the Team rather than directing and controlling. Vodde commented, "A ScrumMaster is not a project manager. The project manager role within Scrum ceases to exist as its responsibilities are moved to the other Scrum roles."[2] Some Scrum projected are reported to have both a ScrumMaster and a project manager (and larger projects using a Scrum of Scrums approach might have a program manager working with multiple ScrumMasters).

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 Product Owner may represent multiple customer constituencies but has the responsibility and authority to reconcile conflicting requirements and determine the business value associated with each item in the Product Backlog.

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

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, Sutherland07, Sutherland09], although large and distributed projects are quite different from the environment assumed for agile methods in general.

[2] B. Vodde, email to Mark Paulk entitled "Re: empirical research into Scrum adoption," 20 May 2009.

Other Engineering Practices

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].

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

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

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

and corollary practices:

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

XP and Scrum are a popular hybrid agile method. XP practices that may be particularly pertinent to successful Scrum adoption include test-driven development [Erdogmus05, Kniberg07, Williams03], refactoring [Fowler99], pair programming [Williams02], and sustainable pace [DeMarco01, Goldratt97].

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

Boehm and Turner identified five critical factors in determining whether agile methods or a plan-driven method was likely to be more suitable for a project: size, criticality, dynamism, personnel, and culture [Boehm04]. Relying on tacit knowledge makes agile methods suitable for small teams but limits scalability. Agile methods may be inappropriate for life-critical and essential moneys projects to the degree they oversimplify design and lack documentation. Agile methods are designed for highly dynamic environments where simple design and refactoring can excel. In general, agile methods require competent people throughout their life. Agile methods are well adapted for cultures with many degrees of freedom.

The Agile Culture

As Schwaber commented in the introduction to this section, Scrum and the other agile methods should not be viewed as a collection of practices, but rather as a culture or a set of values. Most of the agile gurus that I have talked with over the years have emphasized the cultural aspect of agility over a mechanistic set of practices, as did some of those who commented on this research plan.

From an empirical research perspective, this poses a challenge. If we cannot capture an understanding of what an agile project is by asking about what they do, then how can we operationally define in a repeatable way what "agile" or "Scrum" means? After all, if an iteration can be six months, so long as it captures the agile values (in some sense), can one make a

meaningful distinction between agile and traditional waterfall methods? This extreme interpretation of what the experts say, however, makes the wrong emphasis in my judgment. What the agile experts seem to be really saying is that a project can implement all of the agile practices it likes – if it does not adopt the agile values, those practices will become an ineffective façade.

Cohn, however, emphasizes that team and project context trump all other considerations, therefore instead of best practices, what we need to know are good practices and the contexts in which they are successful [Kniberg07]. We might infer that, even if a practice is not "agile" (e.g., six-month iterations), it may be appropriate for the team and project context. We will therefore attempt to probe into agile practices while retaining a sensitivity to agile culture that may spark additional research in the future.

One statement of the underlying principles of agile methods was published in association with the Agile Manifesto. The twelve principles[3] are:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

4. Business people and developers must work together daily throughout the project.

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

7. Working software is the primary measure of progress.

8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

9. Continuous attention to technical excellence and good design enhances agility.

10. Simplicity – the art of maximizing the amount of work not done – is essential.

11. The best architectures, requirements, and designs emerge from self-organizing teams.

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

[3] "Principles Behind the Agile Manifesto," <URL: http://www.agilemanifesto.org/principles.html>, visited 16 July 2009.

DEFINING SUCCESS

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).

Objective measures of cost and benefit would be preferred, and will be collected to the degree available, but rigorous measures of value are frequently lacking in industry projects, so this first round of research will be based on the perception of value.

Shenhar, Levy, and Dvir identified four underlying dimensions for project success [Shenhar97]:

- customer satisfaction, i.e., meeting requirements, fulfilling customer needs
- budget and schedule, i.e., meeting time and budget goals
- business success, i.e., the level of commercial success or market share
- future potential, i.e., opening new markets, developing new technologies

The traditional dimension of meeting time, budget, and performance goals is not really a homogenous dimension, since meeting project resource constraints (time and budget) is one thing, while meeting requirements is another. Poor requirements may result in dissatisfied customers even when specifications are fully met.

During project execution, project managers focus on please prospective customers, meeting time and budget goals, and succeeding commercially (to some extent). After the project is fielded, success is determined more by the impact of the project on the future of the organization. It may be that Microsoft has never had a project come in on budget or on schedule, but they are without question a highly successful company.

In general, agile methods seem to focus well on customer satisfaction and business success. Future potential depends on the organizational reason for initiating the project, so may be an issue outside of project scope. Budget and schedule issues would seem to be secondary, subordinate to customer satisfaction and business success, although important to the degree that the customer and the business prioritize cost and schedule issues.

Treacy and Wiersma identified three value disciplines that companies can follow to be the best in their markets [Treacy97]. These are driven by quite different strategies in satisfying (or delighting customers; for all three customer value is the ultimate measure one's work performance and improving value is the measure of one's success. The three value disciplines are operational excellence, product leadership, and customer intimacy.

Operationally excellent companies provide middle-of-the-market products at the best price with the least inconvenience – the best total cost for their customers. They use standardized assets and efficient operating procedures and reject variety because it burdens the business with cost. They have three advantages: (1) focus, hassle-free basic service a key part of their value proposition; (2) efficient, zero-defect service; and (3) use of information technology to redesign basic service

tasks. Operationally excellent companies live or die by process improvement, governed generally by the principles of total quality management.

Product leaders offer products that push performance boundaries. They offer the best product, and competition is about product performance. Product leaders avoid bureaucracy at all cost. Success is driven by the extraordinary talents of key individuals who develop and market breakthrough after breakthrough. Operating procedures and processes are designed to play into the drivers of individual behavior, which include a thirst for problem-solving and a distaste for bureaucracy. Product leaders create flexible organizational structures and robust processes that enable people to flex their minds without creating disruption. They provide efficient coordination, while accommodating inventiveness and discipline. Product leaders create business structures that don't oppress and stress procedures where it pays the biggest dividend, typically during the final leg of the product development effort.

Customer-intimate companies focus on delivering what specific customers want – the best total solution. They do not pursue one-time transactions; they cultivate relationships. They prefer steady, controlled, incremental evolution of product coupled with expertise that leads the clients through changes in their application and management.

Each value discipline requires a company to emphasize different processes, to create different business structures, and to gear management systems differently. Treacy and Wiersma argue that, if you decide to play an average game, to dabble in all areas, you can't expect to become a market leader. They also argue that you must focus on a specific value discipline while maintaining threshold standards on the other dimensions of value.

Agile methods such as Scrum are well suited to the needs of product-leader and customer-intimate companies. They are useful to operationally excellent companies, but there is more likely to be a culture clash with the existing practices designed to focus on operational excellence. (Best practice frameworks such as CMMI are focused on operational excellence issues.)

Kaplan and Norton defined the Balanced Scorecard (BSC) as a tool to translate an organization's business objectives into a "balanced" set of performance measures in four key areas: financial, customer, internal business processes, and learning and growth [Kaplan96]. These different perspectives allow for a balance between short- and long-term objectives, and leading and lagging measures. The BSC is a management and measurement system that acts as a link between a company's strategic intent and its operational measurements. The benefit of Balanced Scorecard is that it combines the management and measurement systems, helping to keep line management decision-making consistent with corporate strategy. Using the Balanced Scorecard as a management tool results in an organization-wide understanding of objectives, strategies, and BSC measures and results, as well as regular reviews and discussions of the BSC by the organization's leaders and customers.

As an evolutionary approach to software development, agile methods in general appear to address the learning and growth class of concerns in a narrowly focused, project-specific manner, as well as the customer-focused category. The financial category would seem to be a secondary issue, while the internal business process category would appear to be of little direct relation.

CHANGE MANAGEMENT AND SCRUM ADOPTION

Change is not necessarily for the better. Fads and fashions drive change based on imitating other, successful organizations [Abrahamson91, DiMaggio83]. This kind of diffusion occurs when organizations have unclear goals and high uncertainty about the technical efficiency of the innovations they are considering. Organizations that lack a good understanding of the underlying culture of Scrum may adopt an ineffective variant of the method, thus the need to understand more about how Scrum has been implemented before drawing conclusions about the adoption of the method. If we can assume that adoption is based on rational choice, then innovations such as Scrum diffuse when they benefit organizations adopting them, and they disappear when they do not.

Focusing on a rational and efficient-choice style of adoption, the diffusion of innovation literature identifies five factors that affect the successful adoption of an innovation [Rogers83]:

- perceived relative advantage - the extent to which adopters believe the innovation is better than current practice;
- compatibility - the degree to which an innovation is perceived by the adopter as consistent with their needs, values, and experiences;
- simplicity - the degree to which the innovation is perceived as understandable and implementable;
- trialability - the degree to which an innovation can be experimented with on a limited basis; and
- observability - the degree to which an innovation and its benefits can be observed by the potential adopter.

Diffusion of innovation models stress the importance of similarity, or homophily, which is defined as the degree to which innovator and potential adopter share attributes such as objectives, strategies, norms, beliefs, experiences, and cultures.

A common model for characterizing the classes of people involved with technology adoption is that they can be listed as innovators (techies), early adopters (visionaries), early majority (pragmatists), late majority (conservatives), and laggards (skeptics) [Rogers03]. Moore extends this by identifying a “chasm” that separates early adopters and the early majority; a gap between two fundamentally separate phases in the development of a high-tech market [Moore91]. The early phase builds from a few, highly visible, visionary customers, but transitioning to the mainstream phase, where the buying decisions fall predominantly to pragmatists, is a major challenge. The key to Moore’s insight is characterizing the differences between these communities and how to proactively deal with them.

The “chasm” has implications for adoption of models and standards, since a number of enabling mechanisms expedite adoption, including the existence of:

- ownership: agencies responsible for developing and maintaining the best practice framework
- user groups
- conferences and publications, including case studies of adoption and improvement
- training materials: books, continuing education courses, videos

- Web page: for the sponsor of the work and supporting materials
- penetration: breadth of adoption (world-wide vs national or regional)

Fichman and Kemerer take a slightly different perspective by examining the "assimilation gap" between a new technology being acquired by an organization, the traditional mechanism for measuring adoption, and its actual deployment and use [Fichman93; Fichman95]. Many researchers treat the acquisition of a technology as the adoption event, yet the failure to address the actual deployment makes a critical assumption about the last stages of the standard technology adoption curve.

Fichman and Kemerer point out that widespread acquisition of a technology is not necessarily followed by widespread deployment and use, which they characterize as an "assimilation gap." Traditionally, innovation attributes such as relative advantage, complexity, and compatibility are viewed as the determinants of the rate and level of diffusion. Fichman and Kemerer propose that acquisition and deployment have different drivers, even though they are related processes. Acquisition is driven by the expectation of future benefits owing to increasing returns, but knowledge barriers impede deployment.

To address the assimilation gap via organizational learning (or process management) implies that the organization recognizes the difference between acquiring and deploying a technology and is proactive in tracking and addressing deployment issues. This means understanding the factors influencing returns to adoption (such as network externalities, learning-by-doing, and technological interrelatedness) and knowledge barriers (such as complexity and scaling).

Daghfous and White integrated a number of time-based approaches to characterizing the process of innovation that consider product and process evolution and marketing [Daghfous91]. They add an information axis and a focus on how information interacts with the demand and supply axes. Their innovation analysis model has three dimensions – product/process, application linkage, and information.

The product/process axis, also known as the supply axis, is the axis along which events proceed technically, from initial invention to successful innovation. The applications linkage axis, also known as the demand axis, is the axis along which those events that define markets proceed, from initial definition of concept value to successful application of the innovative product. The information axis deals with the transformation of uncertainty and ignorance into precise knowledge. *Uncertainty* means that the information does not exist to remove variance of expectations. *Ignorance* means the information is known or accessible elsewhere, but the innovator is oblivious and thus at competitive disadvantage.

Managerial decisions regarding innovation are dominated by (the lack of) information. Lack of information is a major inhibitor to innovation, and addressing this lack is a direct consequence of innovation – although resolving uncertainty and ignorance requires different approaches. Information gathering along the product/process axis usually results in removing ignorance. Information gathering along the application linkage axis usually results in removing uncertainty. It can be assumed that once information is learned, it will not be forgotten. The ultimate objective, or "bulls-eye," for an innovation is continuing market evolution under precise knowledge, with optimum products from optimum processes satisfying optimum demand. The sequence, if not the timing, of events is predictable using the Daghfous & White model.

To manage innovations that may be adopted externally via organizational learning (or process management) implies that the organization recognizes the importance of the information axis and how it impacts the other axes. This means analyzing exactly where a technology (or product) is on the process/product and application linkage axes.

Beer, Eisenstat, and Spector characterize company-wide change programs as the "fallacy of programmatic change," suggesting that successful transformations usually start at the periphery of the company to solve concrete business problems [Beer 1990]. They identify six steps on the critical path to successful change:

- Mobilize commitment to change through joint diagnosis of business problems.
- Develop a shared vision of how to organize and manage for competitiveness. They suggest that the root of the problems faced by the organization are functional and hierarchical barriers to sharing information and solving problems.
- Foster consensus for the new vision, competence to enact it, and cohesion to move it along.
- Spread revitalization to all departments without pushing it from the top. They suggest that it is better to let each department reinvent the wheel if necessary in finding its own way to the new organization.
- Institutionalize revitalization through formal policies, systems, and structures. Mechanisms such as policies are important so the process continues even after the sponsoring managers of change have moved on to other responsibilities.
- Monitor and adjust strategies in response to problems in the revitalization process. They encourage creating a learning organization to deal with the changing competitive environment.

Pil and MacDuffie argue that radical changes, including fundamental shifts in technologies and methodologies, can be competence destroying [Pil96]. This can lead to "competency traps," where organizations maintain inferior routines they have had favorable experience with in the past. Thus superior practices that do not yield immediate results face a high risk of not being retained. Oddly enough, the cost of change is less for poorly performing organizations.

Pil and MacDuffie identified two major types of disruptions in assembly plants that could result in unfreezing the current way of doing things: major product changeovers and significant new additions to the plants.

High-involvement work practices may represent "competence-destroying" change, which is difficult to implement, and may lead to worsened performance in the short term (and thus *not* an economically rational choice for individual managers held accountable for short-term results). These practices may also have a less favorable impact on performance if they are not given adequate time to develop. For both of these reasons, firms may be discouraged from making changes in work practices (particularly change involving "bundles" of interdependent practices

rather than individual practices), or from continuing with change efforts beyond an initial trial period.

Given these impediments to change, Pil and MacDuffie argue that there are three key factors at the plant or establishment level that drive the adoption of new work practices (and "bundles" of practices): (1) the level of complementary organizational practices and technologies that would increase the *benefit* from the new practices, (2) the performance levels the organization is achieving with its current practices, and (3) organizational characteristics or actions that alter the *cost* of introducing the new practices.

IMPROVEMENT FACTORS

Change management covers a broad range of adoption issues. Process improvement addresses a more focused set of issues more directly related to software engineering methodologies, which may provide additional factors to consider for Scrum adoption. Some have observed that the majority of improvement programs fail, with reports of 80% failures being fairly typical [Goodman96] and indications that fewer than 10% of the Fortune 1000 have well-developed TQM programs [Repenning01]. If adoption failures for agile methods are as high, it would perhaps not be surprising.

El Emam, Goldenson, McCurley, and Herbsleb observed that the most important factor in distinguishing between success and failure of software process improvement efforts is the extent to which the organization is focused in its improvement effort, with clearly defined goals and consistent directions set by senior management [Emam98]. Factors that may be worth exploring for Scrum adoption include:

- *Lack of management commitment*. Goodman believes this problem occurs because the people involved in the quality programs talk quality rather than business in terms that managers relate to [Dyba05, Goodman96, Kasse00, Niazi06, Powell95]. Discussing Scrum in engineering terms rather than business concerns could lead to a similar problem.

- *Lack of clearly defined goals*. Goals depend on a clear statement of the desired benefit to be obtained by adopting a new technology, which provides a foundation for measuring progress and determining success [Dyba05, Emam98, Kasse00].

- *Staff inexperience.* The skills needed to solve technical problems can be very different from the skills necessary to successfully manage people [Baddoo03, Goodman96, Kasse00, Niazi06], especially when changing management paradigms from control-oriented approach to an empowered, coaching style.

- *Lack of training*. Investing in the necessary training to enable new methods and techniques to flourish is a common problem [Niazi06, Powell95]. Agile methods such as Scrum require new skills in management, technical, and teamwork areas.

- *The Pilot Syndrome*. The effects of pilots may take many months to assess and any benefits they deliver will be confined to the pilot area and will not impact the overall business [Goodman96]. Management commitment may slip away during the pilot. The delay between investing in improvement activities and reaping the benefits is a general problem with working smarter versus working harder [Repenning01].

- *A lack of measurement*. Evidence-based management supports the adoption of successful new technologies based on objective evidence [Dyba05, Niazi06, Powell95, Rousseau07, Schaffer92]. Agile methods, however, are not known for their use of data in decision making, including quantitative or statistical arguments for why to adopt agile methods.

- *Process versus results orientation*. Arguments in favor of a new technology should be based on the business results obtained to get senior management buy-in [Schaffer92]. This goes back to the definition of success, but observed results are crucial to successful adoption.

- *Commercial pressures*. External pressures, even if perhaps unrealistic, can lead to failure [Baddoo03, Kasse00]. The Product Owner is responsible for prioritizing conflicting requirements, and perhaps even deciding to terminate the project if it cannot meet its business objectives, but it is human nature to succumb to pressure on occasion.

- *Tool support*. While technology is not a silver bullet, it is important to support the efficient adoption of new methods [Kasse00]. Automated regression testing, for example, makes the adoption of agile methods much easier.

CULTURAL FACTORS AFFECTING CHANGE MANAGEMENT

For this initial round of research, cultural issues affecting adoption will not be investigated, but they are important factors influencing how people work together, deal with change, and feel about innovation. Some consideration of national and organizational cultures is necessary to understand Scrum implementation and adoption issues, but a full-blown study of cultural factors is beyond the scope of this initial investigation.

Hofstede identified four largely independent dimensions of differences among national value systems [Hofstede96]. These were labeled power distance (power is distributed unequally, status differences), uncertainty avoidance (tolerance for uncertainty and ambiguity), individualism vs. collectivism (interests of the individual vs. interests of the group that individual is a member of), and masculinity vs. femininity (confrontation vs. compromise). A fifth dimension identified later was termed Confucian dynamism (long-term vs. short-term orientation in life and work). We might expect agile methods to flourish in a culture with small power distance, willingness to tolerate ambiguity, willingness to compromise, and a long-term view.

Constantine defined four broad categories of organizational culture [Constantine93, Constantine95]. Closed paradigm organizations are hierarchical. They rely on standards and rules of operation to promote continuity. Random paradigm organizations directed at innovation and change through individual creativity. Open paradigm organizations rely on open communication and consensual decision making. Synchronous paradigm organization use tacit agreement for alignment. Each paradigm has particular strengths, as well as intrinsic weaknesses. For teams, Constantine recommends the structured open team, which is a tight-knit, closely integrated team of professional equals with clear differentiation of functions only as necessary for effective functioning. To avoid problems intrinsic to the consensual decision making, the technical leader is responsible for resolving technical disputes.

There are a number of other national and organizational culture taxonomies that might provide useful insight [Handy91, Schein92], but the Hofstede and Constantine models will be the initial focus of any cultural analyses we may undertake in future research.

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 target groups, 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 may be desirable 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 address 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.

At the Agile 2009 conference, Larry Maccherone was approached about the possibility of further research in agile methods based on this investigation of Scrum adoption issues. This will be considered at an appropriate time.

REFERENCES

Abrahamson91 E. Abrahamson, "Managerial Fads and Fashions: The Diffusion and Rejection of Innovations," The Academy of Management Review, Vol. 16, No. 3, July 1991, pp. 586-612.

Baddoo03 N. Baddoo and T. Hall, "De-motivators for Software Process Improvement: An Analysis of Practitioners' Views," The Journal of Systems and Software, Vol. 66, No. 1, April 2003, pp. 23–33.

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.

Beer90 M. Beer, R.A. Eisenstat, and B. Spector, "Why Change Programs Don't Produce Change," Harvard Business Review, Vol. 68, No. 6, November/December 1990, pp. 158-166.

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.

Boehm02 B.W. Boehm, "Get Ready for Agile Methods, With Care," IEEE Computer, January 2002, pp. 64-69.

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.

Constantine93 L.L. Constantine, "Work Organization: Paradigms for Project Management and Organization," Communications of the ACM, Vol. 36, No. 10, October 1993, pp. 35-43.

Constantine95 L.L. Constantine, *Constantine on Peopleware*, Yourdon Press, Englewood Cliffs, NJ, 1995.

Daghfous91 A. Daghfous and G.R. White, "Information and Innovation: A Comprehensive Representation," University of Pittsburgh, Department of Industrial Engineering, Technical Report 91-4, 1991.

DeMarco01 T. DeMarco, *Slack: Getting Past Burnout, Busywork, and the Myth of Total Efficiency*, Broadway Books, New York, 2001.

DeMarco02 T. DeMarco and B.W. Boehm, "The Agile Methods Fray," IEEE Computer, June 2002, pp. 90-92.

DiMaggio83 P.J. DiMaggio and Walter W. Powell, "The Iron Cage Revisited: Institutional Isomorphism and Collective Rationality in Organizational Fields," American Sociological Review, April 1983, pp. 147-160.

Dinakar09 K. Dinakar, "Agile Development: Overcoming a Short-Term Focus in Implementing Best Practices," *OOPSLA'09*, Orlando, October 2009.

Dyba05 T. Dyba, "An Empirical Investigation of the Key Factors for Success in Software Process Improvement," IEEE Transactions on Software Engineering, Vol. 31, No. 5, May 2005, pp. 410-424.

Emam98 K. El Emam, D. Goldenson, J. McCurley, and J. Herbsleb, "Success or Failure? Modeling the Likelihood of Software Process Improvement," International Software Engineering Research Network, ISERN-98-15, August 1998.

Erdogmus05 H. Erdogmus, M. Morisio, and M. Torchiano, "On the Effectiveness of the Test-First Approach to Programming," IEEE Transaction on Software Engineering, Vol. 31, No. 3, March 2005, pp. 226-237.

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.

Fowler99 M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts, *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, Boston, 1999.

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.

Goodman96 P. Goodman, "The Practical Implementation of Process Improvement Initiatives," Chapter 13 in *Software Quality Assurance and Measurement: A Worldwide Perspective*, N.E. Fenton, R. Whitty, and Y. Lizuka (eds), 1996, pp. 148-156.

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.

Kasse00 T. Kasse and P.A. McQuaid, "Factors Affecting Process Improvement Initiatives," Crosstalk: the Journal of Defense Software Engineering, Vol. 13, No. 8, August 2000, pp. 4-8.

Keefer03 G. Keefer, "Extreme Programming Considered Harmful for Reliable Software Development 2.0," <URL: http://avocallc.com/downloads/ExtremeProgramming.pdf>, September 2003.

Kniberg07 H. Kniberg, *Scrum and XP from the Trenches*, C4Media, InfoQ Enterprise Software Development Series, 2007.

Larman04 C. Larman, *Agile and Iterative Development: A Manager's Guide*, Addison-Wesley, Boston, 2004.

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.

Paulk01 M.C. Paulk, "Extreme Programming from a CMM Perspective," IEEE Software, November/December 2001, pp. 19-26.

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.

Powell95 T.C. Powell, “Total Quality Management as Competitive Advantage: A Review and Empirical Study” Strategic Management Journal, 16, 1, January 1995, 15-37.

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.

Rakitin01 S.R. Rakitin, “Manifesto Elicits Cynicism,” IEEE Computer, Vol. 34, No. 12, December 2001, p. 7.

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.

Repenning01 N.P. Repenning and J.D. Sterman, “Nobody Ever Gets Credit for Fixing Problems that Never Happened: Creating and Sustaining Process Improvement,” California Management Review, Vol. 43, No. 4, Summer 2001, pp. 64-88.

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.

Skowronski04 V. Skowronski, "Do Agile Methods Marginalize Problem Solvers?" IEEE Computer, October 2004, pp. 120, 118-119.

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.

Sutherland07 J. Sutherland, A. Viktorov, J. Blount, and N. Puntikov, "Distributed Scrum: Agile Project Management with Outsourced Development Teams" *40th Hawaii International Conference on System Sciences*, Big Island, Hawaii, January 2007.

Sutherland08 J. Sutherland and B. Vodde, "The Nokia Test, Extended with Scoring by Jeff Sutherland," <URL: http://www.cedur.se/nokia_test2.html>.

Sutherland09 J. Sutherland, G. Schoonheim, and M. Rijk "Fully Distributed Scrum: Replicating Local Productivity and Quality with Offshore Teams," *42nd Hawaii International Conference on System Sciences*, Waikoloa, Hawaii, January 2009.

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.

Williams02 L.A. Williams and R.R. Kessler, *Pair Programming Illuminated*, Addison-Wesley, Boston, 2002.

Williams03 L.A. Williams, E.M. Maximilien, and M. Vouk, “Test-Driven Development as a Defect-Reduction Practice,” *The 14th International Symposium on Software Reliability Engineering*, November 2003, pp. 34-45.

APPENDIX A. QUESTIONS FOR A COMPREHENSIVE SURVEY

The following set of questions is a rough draft of the content that we might wish to include in the Scrum adoption survey. The actual survey will need to be much shorter, for usability reasons, than this, and the questions and answer scales will need to be fleshed out. Appendix B contains the draft survey, thus we have three different forms of the information we would ideally like to collect – that described in the body of this report, filtered to the items described roughly in Appendix A, then filtered again to create a usable instrument in Appendix B. Note that in this survey the unit of analysis is the project. Respondents are assumed to be ScrumMasters, and respondents may enter multiple responses for multiple projects.

Respondent Demographics

What is your email address?

What certifications do you have? Certified ScrumMaster? ASQ Certified Software Quality Engineer? IEEE Certified Software Development Professional?

Project Demographics

How many months has the Scrum project been running?

How many people are on the Development Team?

What kind of application is the Scrum project building (e.g., Web, MIS, embedded system)? Is Scrum being used on a non-software project?

Organizational Demographics

What is the company name? The organizational (business unit) name?

How big is the organization (number of employees)?

Is the organization ISO 9001 certified? CMMI continuous representation? CMMI level n? ISO/IEC 15504 / SPICE level n?

Scrum Practices

Is there a single ScrumMaster? Is he or she a Certified ScrumMaster? Is there still 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? Does the Product Owner have the responsibility and authority to

determine the business value for the work done by the project? Is the Product Owner co-located with 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? Are there part-time members on the Development Team? Do team members sit together where they can easily see and hear their colleagues? For distributed teams, are different teams separated or are team members within a team separated?

How much requirements volatility (i.e., new or changed user stories in the Product Backlog) is there on this project? On average, less than 1% per month? 3%? 5%? 7%? 10%? More than 10% changes per month?

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?

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? Does the Product Owner set the priority for the user stories? Does the Development Team set the estimates for the user stories? 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

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 for the project? Are risks reviewed at the end of each Sprint?

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

Agile Culture

Is there an open, cooperative, and collaborative relationship with the customer (Product Owner)?

Is there an open, cooperative, and collaborative relationship within the Development Team?

Is the Scrum method that you are using an effective and efficient way of building software?

Are possible improvements to the Scrum method being thoughtfully considered and adopted where they add value?

Is there an appropriate balance between working software and documentation?

Defining Success

Is there a “product vision” that clearly states the business goals and requirements for the product? That is, the financial and marketplace goals, such as profit, rate of return, and market share?

Is addressing customer needs the highest business priorities for the project? Which is a more important priority, budget / schedule constraints or new features and innovation?

Change Management

Is Scrum considered distinctly superior to other methods that may have been used by the organization in terms of business objectives?

Are Scrum and the “agile culture” considered compatible with the existing organizational culture?

Do people in the organization consider themselves to understand Scrum and how it should be implemented? Is Scrum considered radically different from prior ways of building software?

Has Scrum been piloted? Is the current Scrum work a pilot effort? Has Scrum been piloted successfully but not adopted more widely?

Are you aware of the Scrum Alliance or other user groups to support agile adoption? Are you a member of one of these groups?

Are you aware of conferences such as the Scrum Gathering to support Scrum users? Have you ever attended one of these conferences? Do you attend them on a regular basis?

Have all of the people in the Scrum projects been trained in how to implement Scrum?

Process Improvement Factors

Is senior management actively sponsoring the piloting and/or adoption of Scrum?

Has senior management stated a clear set of business goals to be achieved in adopting Scrum? Have measures been identified to use in monitoring progress in achieving those goals?

Do you, as the ScrumMaster, have the skills and training necessary to implement the Scrum method?

Does the Product Owner have the skills and training necessary to implement the Scrum method?

Does the Development Team have the skills and training necessary to implement the Scrum method?

Does the Development Team have appropriate and effective tools for doing project work?

Cultural Factors

Do you consider the Development Team to be appropriately empowered to do its work using the Scrum method?

Do you consider the decision making style of the Scrum project to be consensus-based?