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. In this research the author investigated the issues associated with Scrum adoption: the practices that characterize the Scrum method, barriers and enablers for successful adoption of Scrum, and the perceived value of Scrum.

INTRODUCTION

This article describes an empirical software engineering research project into Scrum adoption. As arguably the most popular of the agile methods at this time, Scrum is a reasonable focus, yet it is one of a number of methods, whether characterized as agile or not, based on an incremental and iterative lifecycle. Focusing on Scrum allowed the author to investigate specific aspects of adopting the method, but he has also attempted to consider some of the broader issues without expanding the scope of the survey so much as to affect usability.

Potential barriers and enablers of successful technology adoption were considered to identify what issues might be of interest in the case of Scrum. Authorities from the Scrum community, for example, Ken Schwaber and Jeff Sutherland, were consulted to identify what is, and is not, a reasonable tailoring of Scrum. Before investigating the factors associated with Scrum adoption, one must be sure he or she is 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?

The survey also identifies factors that 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 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.
of the interesting questions about factors affecting Scrum adoption include:

- What external factors, such as access to user groups, affect Scrum adoption?
- What internal factors, such as training, affect Scrum adoption?

The 2011 Scrum adoption survey is described in more detail at mark.paulk123.com/mark_files/Papers/sas11.html including the sanitized survey responses. A number of prior investigations of agile methods and the factors affecting their success are described in the online supplement to this article, along with the factors identified in a variety of change management, technology adoption, and diffusion of innovations studies.

SCRUN 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 Schwaber and Sutherland, but these practices must be interpreted in light of agile values and principles. Some of the practices described in this section are not universally considered part of Scrum and are noted as appropriate.

Elaborating this point, Schwaber (2009) 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. Cockburn characterizes the sweet spots for agile projects as two to eight people in one room, on-site usage experts, one-month increments, fully automated regression tests, and experienced developers (Cockburn 2002). High requirements volatility is usually assumed for agile projects.

Schwaber’s two books (Schwaber and Beedle 2002; Schwaber 2004) can be considered the foundational statements of what Scrum is, although Schwaber and Sutherland (2011) have published a “Scrum Guide” that may be considered the formal definition of the method. These descriptions of Scrum and its practices are elaborated and clarified in various reports and training, as well as related books (Cohn 2005; Larman and Vodde 2008). The following brief description of Scrum practices and roles highlights the specific aspects of Scrum that one may wish to investigate to verify that a Scrum implementation is valid.

Inappropriate Scrum variations are colloquially known as “ScrumButs.” 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 Scrum. Scrum is frequently described in terms of:

- Three roles: ScrumMaster, product owner, and development team
1. What has been done since the last daily Scrum.
2. What will be done before the next daily Scrum.
3. What obstacles are in the way.

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 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. (Note that the new Scrum Guide [Schwaber and Sutherland 2011] uses "ordered" rather than "prioritized".)

The sprint backlog is an output of the sprint planning meeting. It consists 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.

A burndown chart graphs the estimated work remaining (measured in story points) against time. The sprint backlog burndown is a graph of the amount of sprint backlog work remaining in a sprint against time. The release burndown graph records the sum of remaining product backlog estimated effort against time.

A number of other ceremonies and artifacts are usually included in Scrum. 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 Scrum only talks about release planning rather than a release plan. He suggested that a release plan is usually just a prioritized, roughly estimated product backlog, with stories roughly assigned to the next few iterations. A release plan is explained in several Scrum descriptions (Cohn 2005; Schwaber and Sutherland 2011), but not all (Schwaber and Beedle 2002; Schwaber 2004).
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.

Scrum projects are typically small to medium sized, but large Scrum projects have been reported. These are typically organized as a “Scrum of Scrums” (Schwaber 2004). Scrum has also been adopted at the enterprise level (Schwaber 2007). Agile methods in general are assumed to be geographically co-located, but distributed (virtual) teams have been described (Ramesh et al. 2006; Sutherland et al. 2007; Sutherland, Schoonheim, and Rijk 2009), although large and distributed projects are quite different from the environment assumed for agile methods in general.

SURVEY DESIGN

A Web-based survey was used to solicit information on Scrum implementation and adoption. Web-based surveys are convenient, allow rapid data collection, are cost effective, allow respondents ample time to consider their responses, can be confidential and secure, and allow the researcher to target specialized populations (Rea and Parker 2005).

The general population targeted by this survey could be considered the universe of software professionals, as well as the customers and the users of software products. Three communities provide a reasonably comprehensive sample of the overall software community, emphasizing quite different perspectives on software engineering: the ScrumAlliance, Project Management Institute (PMI) members, and high-maturity organizations.

The membership of the Scrum Alliance provides a sample of certified ScrumMasters and other Scrum roles, but this is not an ideal sample since one of the objectives of this research is to understand the barriers to Scrum adoption. Members of the Scrum Alliance are not representative of the larger population of software professionals since there is a reasonable assumption that they have overcome many of the barriers to adoption that the author would like to understand better. The Scrum Alliance sample should be useful in understanding the practices of successful Scrum projects, but additional respondents are needed to address the self-selection bias inherent in membership. As of August 22, 2011, the Scrum Alliance had more than 90,000 members, according to its website.

Membership in the PMI includes thousands of Certified Project Management Professionals, many of whom manage software projects. A PMI sample provides a view into organizations that have decided not to adopt Scrum or other agile methods at this time. As of August 22, 2011, the PMI Information Systems Community of Practice (COP) had 707 members; the PMI Agile COP had 1,264 members; the combined COPs had 1,640 distinct members.

High-maturity organizations, as measured using the Software Engineering Institute's Capability Maturity Model Integration (CMMI) for Development (Chrissis, Konrad, and Shrum 2011), should be actively interested in innovative software engineering concepts that can improve performance. Agile methods are an increasingly popular, if sometimes controversial, topic within the software community. Surveying high-maturity organizations to determine their adoption of agile methods, including Scrum, and their reasons for adopting (or not) provides the perspective from organizations that in principle are near the frontier of adopting effective software engineering innovations, although they may remain in the early (or late) majority in diffusion of innovation terms, depending on their business objectives and environment. As of August 22, 2011, there were 80 maturity level 4 organizations in the SEI's Software Engineering Information Repository; there were 213 maturity level 5 organizations.

This is an observational study. This kind of survey is intrinsically biased by the nature of the respondents. Those interested in responding are more likely to be favorable to agile methods and Scrum than those who do not respond. The results are indicative of the overall software community but not definitive. It does, however, provide insight into the perspectives of the specific communities investigated, as well as the practices implemented by those who have successfully implemented Scrum.

Caution is appropriate in generalizing the results of this survey to the larger software community. Still, useful conclusions can be drawn from the perspective of the subpopulations that were specifically targeted. Many organizations will be able to identify with one or more of these communities.
SURVEY RESPONSES

After sanitizing and cleaning up the survey responses, there were 205 responses to the 2011 Scrum adoption survey, which was broadcast to the three communities identified previously. This is a small response rate, but large enough to be useful. The survey questions are documented at mark.paulk123.com/mark_files/Papers/sas11.html. Note that in some cases multiple responses could be checked; therefore, the numbers do not necessarily sum to 205. Scrum was the most popular method used by the respondents, as shown in Figure 1.

As can be seen in Figure 2, the survey respondents range from having never heard of Scrum before to those who consider Scrum to be the normal way of building software for their organization. Of the 205 respondents, only 105 are deploying and/or using Scrum across the organization.

Figure 3 suggests that, for those organizations adopting Scrum, the telecommunications and financial services sectors are adopting Scrum most aggressively. Aerospace and public utilities are the slowest in adopting Scrum.

As might be expected, Figure 4 indicates that the most common team size for Scrum projects is seven to nine professionals. Teams with more than 16 members are fairly common, suggesting that a team of teams approach is likely.

There have been heated discussions on various online discussion groups about the role of project managers in agile projects, beginning in 2010 and extending to 2012. Some of this discussion may be summarized as project management may be
necessary, but project managers are “evil”—a negative reaction to the popular command-and-control style of project management that clashes with the empowerment and self-organizing teams intrinsic to the agile culture. As Figure 5 indicates, almost as many projects have a project manager/ScrumMaster as have a “pure” ScrumMaster, and many projects have both a project manager and a ScrumMaster. While this survey does not provide any insight into the advisability of agile projects having a project manager, it does highlight the context in which many projects operate.

As shown in Figure 6, most product owners have to deal with integrating the agendas of multiple stakeholders—fewer have the authority as the single point of control for prioritizing the business value of the requirements (user stories). Only about one-third of product owners are co-located with the developers, which can lead to lessened communication between the technical and business perspectives.
building the critical infrastructure and support in the early sprints needed for ongoing work. Practices such as pair programming and measuring technical debt may be useful even if they do not appear to be widely adopted. Practices such as test-driven development and risk management, which are recommended by many software experts as useful extensions to Scrum and other agile methods, are areas where there are significant opportunities even though many projects are using these practices.

Figure 7 suggests that development teams are typically co-located, although one team member may be remotely located (and may be a business analyst interacting with the customer). Distributed teams are fairly common, however. It is worth noting that most development teams work at a sustainable pace. Less than half of development teams are composed of generalists who can perform any task that needs to be done.

As illustrated by Figure 8, the bulk of Scrum projects are dealing with requirements volatility on the order of 5 to 20 percent new, changed, or deleted requirements every month. A handful must deal with requirements volatility greater than 50 percent per month, suggesting that “exploratory” is a reasonable characterization.

Figure 9 shows that the most popular length for a sprint is two weeks. When training Scrum teams, the author recommends that teams begin with two-week sprints because they are short enough to provide rapid feedback, which in turn promotes fast learning, and the length of the sprint can be adjusted as the team learns what works best.

Figure 10 indicates that some controversial engineering practices are occurring in Scrum projects, while some recommended practices are not as widely adopted as one might expect. Given the assumption of requirements volatility, having a requirements specification in many projects is surprising. While not broadly discussed, developing an architecture for many Scrum projects can be considered part of
As can be observed in Figures 11 to 13, compared to other software engineering methods, the quality of Scrum products is higher, the cost tends to be less, and customer satisfaction is notably improved. From a “results” perspective, therefore, Scrum is an effective and efficient software engineering method. From a quality perspective, better product quality and customer satisfaction are associated with Scrum projects.

CONCLUSIONS

The bottom line of this survey is that Scrum is an effective and efficient method that when properly implemented leads to improved results. Some practices are controversial; resolving the controversies should, ideally, be based on evidence rather than opinion. Recommended engineering and management practices—and the correct implementation of core Scrum practices—continue to be challenges worth further study and consideration.

Further insights can be garnered from the 2011 survey data by investigating various perspectives on the responses, including those not summarized in this article. Future surveys on the adoption of agile methods can build on the lessons learned from this research.

The bottom line to all studies of agile adoption at this point in time is that uptake continues to expand and is likely to continue to do so for the foreseeable future. The greatest risk to adoption of agile methods is likely to be misuses and abuses of the methods. The most significant opportunity for continued adoption is adaptation of the methods to environments that differ from the “sweet spot” identified for agile.

ACKNOWLEDGEMENTS

This research was funded by the Scrum Alliance. Thanks also to my colleagues who contributed to this work, including Noopur Davis, Karthik Dinakar, Robin Dymond, Scott Duncan, Javier Garzás, Dennis Goldenson, Larry Maccherone, James McCurley, Ken Schwaber, Jeff Sutherland, Bas Vodde, David Zubrow, and Michael Zuccher.

REFERENCES


Schwaber, K. 2009. Email communication to the author, 27 April.


BIOGRAPHY

Mark Paulk teaches software engineering at the University of Texas at Dallas and is a consultant in software process improvement, high-maturity practices, and agile methods. He was with Carnegie Mellon University from 1987 to 2012. While at the Software Engineering Institute at Carnegie Mellon, he led the work of the Capability Maturity Model for Software. He has contributed to ISO and IEEE standards, notably ISO/IEC 15504, and is a co-author of the eSourcing Capability Model for Service Providers. He is a Fellow of ASQ, an ASQ Certified Software Quality Engineer, and a Senior member of the IEEE. He can be reached by email at Mark.Paulk@utdallas.edu.
APPENDIX A
Factors Affecting Adoption Success

Change is not necessarily for the better. Fads and fashions drive change based on imitating other, successful organizations (Abrahamson 1991; DiMaggio and Powell 1983). 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 one 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 (Rogers 2003):

• 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
Supplement for: A Scrum Adoption Survey

- 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
- Observability: The degree to which an innovation and its benefits can be observed by the potential adopter

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) (Rogers 2003). 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 (Moore 1991). 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)

It appears that Scrum has moved into the early majority adoption phase around the world.

Fichman and Kemerer (1999) 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. 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.

Addressing 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).

Daghfoos and White (1991) integrated a number of time-based approaches to characterizing the process of innovation that consider product and process evolution and marketing. 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 a 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 and 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 (1990) 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. They identify six steps on the critical path to successful change:

1. Mobilize commitment to change through joint diagnosis of business problems.
2. 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.
3. Foster consensus for the new vision, competence to enact it, and cohesion to move it along.
4. 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.
5. 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.
6. 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 (1996) argue that radical changes, including fundamental shifts in technologies and methodologies, can be competence destroying. 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 (1996) 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.

El Emam et al. (1998) 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. 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 (Dybå 2005; Goodman 1996; Kasse and McQuaid 2000; Niazi, Wilson, and Zowghi 2006; Powell 1995). 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 (Dybå 2005; El Emam et al. 1998; Kasse and McQuaid 2000).

- **Staff inexperience.** The skills needed to solve technical problems can be very different from the skills necessary to successfully manage people (Baddoo and Hall 2003; Goodman 1996; Kasse and McQuaid 2000; Niazi, Wilson, and Zowghi 2006), 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 (Niazi, Wilson, and Zowghi 2006; Powell 1995). 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 (Goodman 1996). 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 (Repenning and Sterman 2001).

- **A lack of measurement.** Evidence-based management supports the adoption of successful new technologies based on objective evidence (Dybå 2005; Niazi, Wilson, and Zowghi 2006; Powell 1995; Rousseau and McCarthy 2007). 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. 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 (Baddoo and Hall 2003; Kasse and McQuaid 2000). 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 (Kasse and McQuaid 2000). Automated regression testing, for example, makes the adoption of agile methods much easier.

Hofstede (1996) identified four largely independent dimensions of differences among national value systems. 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 the individual belongs to)
- 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). One 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 (Constantine 1993; Constantine 1995).

- 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 organizations 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.

**APPENDIX B**

**Related Surveys and Research**

There have been a number of prior investigations of agile methods and the factors affecting their success. Related surveys and empirical research can inform the kinds of questions one may wish to explore, as well as setting a context for analyzing the responses to this survey.

VersionOne has done a series of annual surveys on the state of agile methods (VersionOne 2010). In their fifth survey, they had 4,770 participants. The median size of organizations was 80, with 32 percent having 250-plus employees. In 47 percent of the organizations, more than half of the projects were using agile methods, with Scrum being the most popular (58 percent), followed by Scrum/XP hybrids (17 percent) and custom hybrids (5 percent).

Practices adopted by the majority of respondents included:

- Iteration planning (83 percent)
- Daily standup (82 percent)
- Unit testing (77 percent)
- Release planning (72 percent)
- Retrospectives (68 percent)
- Burndown (67 percent)
- Continuous integration (65 percent)
- Automated builds (60 percent)
- Velocity (57 percent)
- Refactoring (57 percent)
- Coding standards (56 percent)

Test-driven development was used by 46 percent, pair programming was used by 33 percent, and kanban was used by 18 percent.

The leading causes of failed agile projects included:

- Lack of experience with agile methods (14 percent)
- Company culture at odds with core agile values (11 percent)
- External pressure to follow traditional waterfall practices (8 percent)
- Lack of management support (7 percent)
- Unwillingness of the team (6 percent)
- Insufficient training (5 percent)

The leading barrier to further agile adoption was the (lack of) ability to change the organizational culture. The greatest concern about adopting agile was loss of management control.

The most common reasons for adopting agile methods were to accelerate time to market, enhance the ability to manage changing priorities, and increase productivity. Sixty-six percent viewed agile as providing a faster time to completion. The most commonly reported benefits from implementing agile were enhanced ability to manage changing priorities, improved project visibility, improved alignment between IT and business objectives, improved team morale, and accelerated time to market.
Supplement for: A Scrum Adoption Survey

Scott Ambler has performed a number of IT-related surveys, including several on various aspects of agile methods (Ambler 2011c); his 2010 survey of the agile state of the art had 180 respondents (Ambler 2010a). The most common iteration length was two weeks (51 percent); 3 percent reported a variable length iteration.

Ambler’s 2010 survey of agile success had 108 respondents (Ambler 2010b): 55 percent of agile projects were perceived as successful, 35 percent as challenged, and 10 percent as failed.

Ambler’s April 2011 survey of agile teams had 82 respondents (Ambler 2011a). The average team size was 10 members, after excluding two outlier teams with more than 100 members. Forty-seven percent of the teams were co-located, and 30 percent were distributed. His October 2011 survey of agile teams had 89 respondents (Ambler 2011b). The most common title for the team’s leadership role was project manager.

Lai, Kongyai, and Börstler (2011) did a survey on adaptation of agile practices with 468 respondents. The most popular type of software application, with 49 percent of their respondents, was Web applications, followed by 23 percent finance, 16 percent healthcare, and 16 percent mobile applications. Scrum was the most popular agile method (56 percent), with XP-Scrum hybrids being used by 15 percent, and XP by 8 percent. Practices used without adaptation or with little adaptation by three-quarters or more of the respondents who had adopted Scrum or an XP-Scrum hybrid included:

- ScrumMasters
- Sprints
- Sprint planning meetings
- Daily Scrum meetings
- Sprint reviews
- Product backlogs
- Sprint backlogs
- Sprint burndown charts

More than three-quarters of XP adopters used small releases, test-driven development, refactoring, coding standards, and continuous integration. The planning game was used by 59 percent of the XP adopters, pair programming by 56 percent, and on-site customer by 55 percent. Inspections were used by 83 percent of feature-driven development (FDD) adopters.

Chow and Cao (2008) surveyed 109 agile projects. They identified 12 possible success factors in the categories organizational, people, process, technical, and project. Three critical success factors were identified:

- Delivery strategy (regular delivery of software, delivering the most important features first)
- Agile software engineering techniques (well-defined coding standards up front, pursuing simple design, rigorous refactoring activities, right amount of documentation, and correct integration testing)
- Team capability (team members with high competence and expertise, team members with great motivation, managers knowledgeable in agile, managers who have adaptive management style, appropriate technical training to team)

Dybå and Dingsoyr (2008) did a systematic review of 36 empirical studies of agile methods. As of 2005, 76 percent of the studies had been done on XP, with Scrum and lean development methods having only one study each. They found that studies frequently did not describe the methods well; issues of bias, validity, and reliability were not always addressed, and methods of data collection and analysis were often not explained well.

Most studies reported that agile development practices are easy to adopt and work well. Studies of XP indicate that successful teams balance a high level of individual autonomy with a high level of team autonomy and corporate responsibility; good interpersonal skills and trust were found to be important characteristic for a successful XP team. Customers were found to be satisfied with the opportunities for feedback and responding to changes, but the role of on-site customer can be stressful and cannot be sustained for a long period.

APPENDIX REFERENCES


VersionOne. 2010. The state of agile development: 5th annual state of agile survey.