SUCCESSFUL IMPLEMENTATION OF AGILE SOFTWARE DEVELOPMENT METHODOLOGY

Abstract

Many organizations are improving both the efficiency and effectiveness of their software development efforts by implementing an agile software development methodology. Agile methodologies including eXtreme programming (XP), Scrum, and Crystal Methods. This article is intended to point out the decisions and actions taken by organizations that have successfully implemented an agile software development methodology. Organizations that want to improve their chances of success should: select the right methodology for their organization, train their staff on the selected methodology while providing them access to external resources, involve upper management in the methodology implementation, and make any necessary changes to their office facilities to support the intense interaction required by agile development.

Keywords: Agile Software Development, Effectiveness of software development, Efficiency of software development, Agile Methodologies, eXtreme programming (XP), Scrum, and Crystal Methods.

Introduction

The growth of the Internet, the integrated global economy, and emerging technologies has altered the basic rules of software engineering. Traditional development methodologies are too cumbersome to react to rapidly changing requirements and the short product cycles demanded by Internet-enabled business. To meet rapidly changing requirements, software developers have developed agile software development methodologies (SDMs). The most widely adopted agile SDMs are Extreme Programming (XP), Scrum, Dynamic Systems Development Method (DSDM), and Adaptive Software Development (ASD). Agile SDMs typically utilize iterative development, prototyping, and templates to deliver high-quality software in today's dynamic business environment.

The purpose of this research project was to document the major implementation factors that can be controlled by organizations. The research data was collected with an online survey instrument from organizations around the world. Invitations to participate in the survey were sent to members of agile methodology user groups and software process improvement groups around the world. Survey responses came from software developers, managers, and business customers from organizations that use agile, traditional, or no development methodology. Responses were also received from respondents in organizations that failed in their attempts to implement an agile SDM.

The major factors impacting agile SDM implementation were training, management involvement, access to external resources, and corporate size. Factors such as using models, having an implementation plan, collocating the development team, and developing software for Internet or intranet use did not impact the implementation of an agile software development methodology.

Agile methodologies

Software development methodologies are the procedures, business rules, processes, and techniques that developers use to develop software [Roberts, T. et al, 1998]. The use of methodologies helps standardize the software development process and improve the predictability of the process and quality of the resulting software. Traditional SDMs emphasized "complete" documentation and collecting a complete set of fixed business requirements before designing and developing any software. Agile SDMs have been developed that have minimal documentation and are adaptive to changing business requirements.

Agile SDMs are characterized by iterative development, constant code integration, and production of demonstration modules and prototypes [Roberts, T. et al, 1998]. By contrast, traditional methodologies are characterized by complete documentation, rigid processes, and completion of the entire analysis and design phases before coding begins [Highsmith, J. 2002]. Agile SDMs are responsive to rapidly changing customer needs and requirements. Small, large, and medium sized organizations in a variety of industries including manufacturing, health care, information technology, and government are implementing agile SDMs to reduce costs, improve productivity and code quality, and compress development schedules [Reifer, D. J. 2002].

Research basis for this paper

This paper presents the results of a research project undertaken to identify the factors that impact the implementation of an agile methodology. A web-based survey instrument was developed and reviewed by a panel of peers for readability and usability. Once the survey was edited based on the comments made by the panel, the instrument was presented to a panel of methodology experts and revised again to incorporate their suggestions. Invitations to take the survey were e-mailed to the officers of agile methodology or software process improvement groups. A total of 112 responses were received from a variety of industries. The respondents had an average of 14.02 years of industry experience. The industry representation of the respondents is detailed in Table 1.

Table 1. Industry demographics

Industry	*Agile SDM*	*Non-Agile SDM*	*No SDM*
Consulting	11	1	1
Education	2	0	1
Government	2	1	1
IT	23	0	9
Financial	9	1	4
Manufacturing	2	0	1
Retail	2	0	1
Telecommunications	1	1	3
Transportation	1	0	0
Utility	0	0	1
Other	11	1	3
No Answer	7	1	10
Total	71	6	35

Selecting the right agile methodology

Different agile methodologies deliver different benefits. To help organizations select the best methodology for their organization, the survey used in this research project attempted to identify which benefits were associated with each of the popular agile methodologies. The survey presented statements about the benefits most frequently associated with using an agile SDM. There were questions asking whether the respondents strongly agreed, agreed, disagreed, strongly disagreed or were neutral about their methodology delivering specific benefits.

The most common benefit from agile methodologies was the ability to handle new and changing business requirements. The ability to deliver software in shorter time windows and to produce software with fewer defects was the next most common benefits. The results of this project's survey answers are detailed in

Table 2. None of the benefits were realized at all of the organizations even though all of the benefits were all received at a majority of the organizations. Implementing an agile methodology does not guarantee that an organization will receive any benefits as no agile methodology is a silver bullet.

Survey respondents were presented with an open ended question asking if any additional benefits not on the survey list had been derived from their agile SDM implementation. Respondents provided 50 responses to this question, 47 of which were listing one or more benefits. The most commonly additional benefits were: better internal and external marketing, process predictability, improved customer relations, standardized product, and elevating an organization's CMMI maturity level. Several of these benefits are extremely significant from a business viewpoint. Elevating a software development organization to a higher CMM level may be critical to marketing software services to the U.S. Department of Defense [Boehm, B, & Sullivan, K. Software 2000]. Improved marketing and customer relations are important to every software development organization.

While it is difficult to compare methodologies, the survey results showed that different agile methodologies provide different benefits than others. A satisfaction score was computed from the research data by assigning numerical values to the survey responses indicating agreement or disagreement on the benefits received by each organization. Every benefit received from a methodology implementation added to the satisfaction score. ASD received the highest benefits score but was only implemented by one organization and thus it is hard to draw conclusions about ASD. XP delivered the most benefits of all the methodologies implemented by more than one organization. Table 3 contains the benefits score of all the methodologies implemented by the surveyed organizations. The economics of calculating the direct financial value of the benefits of implementing an agile methodology are not proven [Boehm, B, & Sullivan, K. 2000].

Table 2. Benefits of agile development methodologies

Benefit	Strongly Agree	Agree	Neutral	Disagre	Strongly Disagree
Handle new requirements	36	33	7	3	3
Deliver in less time	26	29	17	5	4
Produce fewer defects	21	34	10	12	4
Facilitate management	21	23	18	15	3
Reduce staff turnover	8	15	45	10	2
Improve staff morale	17	31	18	10	4
Increase staff competence	12	43	19	6	1

Table 3. Methodology differences

Methodology	Number of Organizations	Benefits Score
Adaptive Software Development	1	31
eXtreme Programming	26	27.8
Feature Driven Development	4	26.5
Scrum	8	25.6
Homegrown agile methodology	34	24.6
DSDM	3	23.7

Access to external resources

Access to external resources including training, periodicals, and methodology user groups improves the chances for a successful agile methodology implementation. The research found that the impact of training and external resources significantly impacted implementation success. XP implementations are facilitated by having a staff member designated as the methodology coach [Lippert, M., Roock, S., Wolf, H., & Zullighoven, H. 2001]. A coach keeps a development project on track by monitoring methodology usages by team members [McBreen, P. 2000 and Noble, J., Marshall, S, Marshall, S., & Biddle, R. 2004]. There is a natural tendency to deviate from the development process [McBreen, P. 2000]. A coach watches for methodology process deviations and reminds developers of the importance of working within methodology processes [Lippert, M. et al 2001]. Development teams that have difficulties with an agile methodology will work off task while continuing to state that they support the methodology [McBreen, P. 2000]. Ideally, the methodology coach should be a member of the team that constantly interacts with the other team members [McBreen, P. 2000].

Silicon and Software Systems (S3) used a methodology coach to facilitate the introduction of an agile SDM [O'Hara, F. 2000]. The coach worked as a mentor to the development team and set up methodology workshops to train team members in using the methodology. Large organizations such as Motorola also use coaches [O'Hara, F. 2000]. A large manufacturing firm successfully implemented a new software development methodology after its first implementation attempt failed [Nelson, K. N., Buche, M., & Nelson, H. J. 2001]. A senior manager championed the methodology and served as a coach to the development teams. This management level coach obtained resources and political support from upper management, obtained developer buy-in, and placed comments in performance reviews about each member's support or lack of support of the methodology.

An external consultant can serve as a methodology coach [Holmberg, L., & Mathiassen, L. 2001]. As an example, Linq, a software development company, brought in an external consultant to act as a coach in a software process improvement initiative. This consultant enabled implementation of a new methodology that was adopted by several internal development groups. All of Linq's development teams eventually adopted the new methodology to gain the consistency that Linq's customers and management desired [Holmberg, L., & Mathiassen, L. 2001].

Management support and involvement

Agile methodologies benefit from management involvement and support. The research found this factor to be statistically significant. This result was predicted in a literature review and can be considered a common sense result. All projects benefit from the support of management and the resources that will be allocated because of the management support.

Updated office facilities

One of the basic tenets of XP is that XP activities require an open workspace environment [Auer, K. & Miller, R. 2002]. Open workspaces facilitate daily standup meetings and enable paired programmers to collaborate with other team members. If team members must leave their work area to talk to someone, they will generally postpone these interactions [Auer, K. & Miller, R. 2002]. Team members may also make verbal announcements or pleas for help from everyone on the team by simply raising their voice in an open workspace environment [Auer, K. & Miller, R. 2002 and Teasley, S. D., & Covi, L. A. 2003]. Functional furniture, climate control, and furniture arrangement are also important for a development group's productivity. For example, pair programmers typically prefer to work at tables instead of desks so they can comfortably sit side-by-side and share a computer and documents [Beck, K. 2000]. According to Cockburn [Cockburn, A. 2002], after a software compiler, a whiteboard ranks second as an enabling device for collaboration and team communication.

One furniture arrangement that supports agile methodologies is the caves and common design, in which the room is laid out with a central common area where pair programmers and groups of more than two can work together. The size of the central common area varies with the size of the development team. The caves and commons approach enables agile SDM deployment [Cockburn, A. 2002]. The common work area is equipped with computers, whiteboards, bulletin boards, and other communications tools and devices that the development team requests [Cockburn, A. 2002]. Caves or private work areas are reserved for private meetings, telephone conversations, or reflection. The caves are located in the area outside the common work area.

Innovative facilities can promote the adoption of an agile SDM [Poole, C., & Huisman, J. W. 2001]. When Iona Technologies implemented an agile SDM, the entire work area was rebuilt. Management converted 20% of individual cubicles into non-partitioned work areas and installed conference tables, mounted whiteboards on the available wall surfaces, and provided flip charts as well as installing a snack service [Poole, C., & Huisman, J. W. 2001]. These changes led to the predicted increase in staff communications and collaborative group work. The personalization of the common work area and the inclusion of decorations and up-to-date technology encouraged the team members to abandon their individual spaces to work collaboratively in the common area [Auer, K. & Miller, R. 2002]. The elimination of individual cubicles facilitated the acceptance of pair programming [Poole, C., & Huisman, J. W. 2001]. Changes in facilities at the physical workplace to support collaborative agile processes send a message to a development team that management is serious about committing time and money to the project [Beck, K. 2000]. Every organization's facilities, personnel policies, and corporate culture are different and may require unique arrangements [Wake, W. C. 2002]. Although XP supports an open work area, variations will work and may be necessary due to restrictions beyond the development team's control.

Communication tools

Non-collocated teams use software tools such as instant messaging (IM) or e-mail to compensate for the lack of casual interactions associated with collocation [Brush, A. J., & Borning, A. 2003 and Teasley, S. D., & Covi, L. A. 2003]. A growing number of industries including financial services, technology, and telecommunications businesses integrate IM as part of their standard business operating procedures [Isaacs, A. et al 2002 , King, J. 2004 and King, J. et al 2003]. Agile development teams frequently take the initiative to install IM software if there is not a corporate-sanctioned alternative [Majchrzak, A. et al 2004 and Baheti, P. et al 2002]. Geographically distributed teams use IM and collaboration software to facilitate communication between members and customers and minimize miscommunication [Herring, C. 2001]. Collaborative software may support synchronous or asynchronous communications. Synchronous communication occurs when both parties are simultaneously involved in the communication. A telephone call or videoconferencing session is an excellent example of synchronous communication. Asynchronous communications may include time gaps between communications and are exemplified by such as e-mail or bulletin board postings [Vonderwell, S. 2003].

Baheti et al [Baheti, P. et al 2002] investigated the effect of providing distributed XP development teams with videoconferencing technology. These researchers assigned programming projects to 37 groups of university students. To provide valid results, a control group of 24 of the teams were collocated and 13 were geographically distributed. The distributed teams were given a videoconferencing tool with multiple cameras and wall-sized image projections. The videoconferencing tool assisted the distributed teams to be as productive as collocated teams and produce code that had the same amount of errors as the code produced by the collocated teams [Baheti, P. et al 2002]. The majority of organizations responding to the survey implemented electronic communication tools such as NetMeeting and Instant Messaging or a specialized intranet site. Table 4 shows the financial commitment made to purchasing communication tools by the companies reached in this research project. A total of 36 organizations installed online communication tools and that 11 of these organizations spent more than $999 and four organizations spent more than $100,000.

Table-4. Electronic communication tools

Question	Answer	Number of responses
Are electronic communication tools such as NetMeeting, Instant Messaging, or a specialized intranet site provided for the use of the development teams?	Yes	36
	No	27
	Don't know	1
Did the development team install their own communication tools?	Yes	27
	No	34
	Don't know	2
How much money was allocated to acquire and implement the electronic communication tools?	None	33
	Up to $999	2
	Between $999 and $9,999	5
	Between $10,000 and $99,999	2
	More than $100,000	4
	Don't know	16

Resistance to agile methodology implementation

Resistance to an organization implementing an agile methodology is not uncommon [Riemenschneider, C. K. et al 2002]. Developers who are accustomed to using traditional SDM processes can experience considerable culture shock adapting to agile procedures [Trepper, C. 2000]. Developers who are accustomed to working alone may also struggle with the intense communications demanded by agile methodologies [Auer, K. & Miller, R. 2002]. The discipline of adhering to an agile methodology may be confining and restrictive to developers who have no experience working within a structured methodology. Some developers may feel that their "artistic freedom" is restricted by the methodology initiatives.

Managers and customers may resist the agile technique pair programming because of increased initial costs [Auer, K. & Miller, R. 2002]. Pair programming increases initial costs because the initial development person-hours are increased by 15%, but these costs are reclaimed quickly by savings from improved code quality, reduced defects, and enhanced team morale [Auer, K. & Miller, R. 2002]. Agile methodologies require frequent, often daily, meetings among development team members that can be costly to conduct. The customer community should be informed about the financial costs and savings of using an agile SDM as well as be trained in the mechanics of the methodology. [Nelson, K. N. 2001].

To overcome resistance to the change of an agile methodology implementation, Disterer [Disterer, G. 2001] recommends using incentives to encourage workers to accept new practices and share new knowledge. Disterer also recommends the involvement of developers in internal and external methodology user groups. Sharing any and all knowledge about the methodology will increase morale and make developers more accepting of the corporate cultural changes that go along with it.

Conclusion

This research project has identified several actions that can be taken to help implement an agile software development methodology. Organizations around the world are implementing these methodologies and reaping the benefits of agility in our rapidly changing business environment. Research survey results from firms in a variety of industries show that taking the follow steps will improve the likelihood of success:

- Pick the right methodology for your organization that will provide the benefits of most value to your organization.
- Provide training on the methodology.
- Hire or contract for a methodology coach
- Obtain management support, commitment, and involvement in the methodology implementation.
- Update the office facilities to support agile development techniques.
- Install communication tools to

References

- [1] Auer, K. & Miller, R. *Extreme Programming Applied*. 2002. Boston: Addison-Wesley.

- [2] Baheti, P., Williams, L., Gehringer, E., Stotts, D., & Smith, J. *Distributed pair programming: Empirical studies and supporting environments*. (Technical Report TR02-010). Chapel Hill: University of North Carolina. 2002.

- [3] Beck, K. *Extreme Programming Explained*. 2000. Boston: Addison-Wesley.

- [4] Boehm, B, & Sullivan, K. Software economics: A roadmap. In A. Finkelstein (Ed.) *The Future of Software Engineering, Special Volume, 22nd International Conference on Software Engineering*, Limerick, Ireland, New York ACM Press. 2000.

- [5] Brush, A. J., & Borning, A. "Today" messages: Lightweight group awareness via email. In *Conference on Human Factors and Computing Systems CHI '03 Extended Abstracts on Human Factors in Computer Systems* (pp. 920-921). New York: ACM Press. 2003.

- [6] Cockburn, A. *Agile Software Development*. 2002. Boston: Addison-Wesley.

- [7] Disterer, G. (2001). Individual and social barriers to knowledge transfer. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34), Vol. 8* (pp. 1-7). Washington, DC: IEEE.

- [8] Herring, C., & Rees, M. Internet-based collaborative software development using Microsoft tools. In *Proceedings of the 5th World MultiConference on Systems, Cybernetics and Informatics, Information Systems Development, Vol. 1* (pp. 162-167). 2001.

- [9] Highsmith, J. *Agile Software Development Ecosystems*. 2002. Addison-Wesley, Boston, MA.

- [10] Holmberg, L., & Mathiassen, L. Survival patterns in fast-moving software organizations. *IEEE Software, 18*(6), 51-55. 2001.

- [11] Isaacs, A., Walendowski, A., Whittaker, S., Schiano, D. J., & Kamm, C. The character, functions, and styles of instant messaging in the workplace. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (pp. 11-20). New York: ACM Press. 2002.

- [12] King, J. Chat gets serious. *Computerworld*, *38*(8), 33-34. 2004.

- [13] Lippert, M., Roock, S., Wolf, H., & Zullighoven, H. XP in complex project settings: Some extensions. In *Proceedings of Extreme Programming Conference 2001. Villasimius, Cagliari, Italy* (pp. 33-37). 2001.

- [14] Majchrzak, A., Malhotra, A., Stamps, J., & Lipnack, J. Can absence make a team grow stronger? *Harvard Business Review, 82*(5), 131-137. 2004.

- [15] McBreen, P. Applying the lessons of eXtreme programming. In *Proceedings of the Technology of Object-Oriented Languages and Systems (TOOLS-34 2000)* (p. 421-430). Los Alamitos, CA: IEEE Computer Society. 2000.

- [16] King, J., Raven, M. E., Kogan, S., Millen, D. R., & Carey, K. Introducing chat into business organizations: Toward an instant messaging maturity model. In *Proceedings of the 2003 International ACM SIGGROUP Conference on Supporting Group Work* (pp. 50-57). New York: ACM Press. 2003.

- [17] Nelson, K. N., Buche, M., & Nelson, H. J. Structural change and change advocacy: A study in becoming a software engineering organization. In *Proceedings of the 34th Hawaii International Conference on System Sciences* (pp. 1-9). Washington, DC: IEEE Computer Society. 2001.

- [18] Noble, J., Marshall, S, Marshall, S., & Biddle, R. Less extreme programming. In *Proceedings of the Sixth Conference on Australian Computing Education, Vol. 30* (pp. 217-226). Darlinghurst, Australia: Australian Computer Society. 2004.

- [19] O'Hara, F. European experiences with software process improvement. In *Proceedings of the 22nd International Conference on Software Engineering* (pp. 635-640). Los Alamitos, CA: IEEE Computer Society. 2000.

- [20] Poole, C., & Huisman, J. W. Using Extreme Programming in a maintenance environment. *IEEE Software, 18*(6), 42-50. 2001.

- [21] Reifer, D. J. How good are agile methods? *IEEE Software, 19*(4), 16-18. 2002.

- [22} Riemenschneider, C. K., Hardgrave, B. C., & Davis, F. D. Explaining software developer acceptance of Methodologies: A comparison of five theoretical models. *IEEE Transactions on Software Engineering, 28*(12), 1135-1145. 2002.

- [23] Roberts, T., Gibson, M., Fields, K., and Rainer, R. Factors That Impact Implementing a System Development
- Methodology. IEEE Transactions on Software Engineering. 24(8), 640-649. 1998.

- [24] Teasley, S. D., & Covi, L. A. Rapid software development through team collocation. *IEEE Transactions on Software Engineering, 28*(7), 671-683. 2003.

- [25] Trepper, C. Continuous process improvement. *Information Week*, 57-65. 2000.

- [26] Vonderwell, S. An examination of asynchronous communication experiences and perspectives of students in an online course: A study. *Internet and Higher Education*, *6*, 77-90. 2003.

- [27] Wake, W. C. *Extreme Programming Explored*. 2002. Boston: Addison-Wesley.