

# Success Factors of Agile Software Development

Subhas C. Misra, Vinod Kumar, and Uma Kumar  
Carleton University, Ottawa, Canada

## Abstract

*Agile software development methodologies have recently gained widespread popularity. The Agile Manifesto states valuing “individuals, and interactions over processes, and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan” (Fowler, 2002). However, little is known about how effective and efficient agile practices are over the traditional methodologies, and what their success factors are. There have been several disparate anecdotal evidences about the success of software development projects using agile methodologies. In this paper we provide a consolidated picture of the different predictors of agile software development success. This is intended to be a review paper. We have also presented a conceptual framework illustrating the relationships between the different predictor variables and agile software development success.*

## Keywords

Agile software, success factors.

## 1. Introduction

*Agile Software Development* is currently an emerging discipline of Software Engineering, constituting of a set of principles initially advocated by a group of seventeen software practitioners, and now practiced by many software professionals. The principles they advocated leading to the emergence of the agile software development philosophy are based on best practices and their previous success and failure experiences with many software development projects regarding what works and what does not. Each of these practitioners had their own different philosophies about how they approached software development. However, all of them advocated close collaboration between software development and business teams, as opposed to silo development by software teams; face-to-face communication, as opposed to over-emphasis on written documentation in projects; frequent delivery of portions of working software, as opposed to final delivery of the complete product at the end; accepting changing requirements by customers, as opposed to defining a fixed set of requirements “cast-in-stone”; and adaptive organizational capability of teams according to changing business requirements (Fowler and Highsmith, 2001; Fowler, 2002).

In February 2001, the advocates of the agile philosophy met together, and prepared the *Manifesto for Agile Software Development*. Following 2001, many other software practitioners were inspired by the philosophy behind agile software development. There were both success and failure stories of projects that followed the agile methodologies. However, most of evidences so far are *anecdotal* (e.g., Ambler 2002a; Elssamadisy 2001; Gittins *et al.*, 2001; Kini *et al.* 2003; Pine, 2001; Schatz *et al.* 2004) in nature.

In this paper we review the parameters that affect the success of projects adopting agile software development methodologies based on previous anecdotal and practical experience stories. We also

present a conceptual framework showing the relationships between the success of agile software development and its predictors.

## 2. Success Factors

### 2.1. Organizational Factors

#### *Customer Commitment*

The Agile Manifesto (Fowler and Highsmith, 2001) advocates customer collaboration as one of the important requirements for successful software development. One of the principles of agile software development is giving highest priority to achieving customer satisfaction through early and continuous delivery of valuable software (Fowler and Highsmith, 2001). This requires that the customers are not only available on site with the software development team, but also highly motivated, active, and consider themselves responsible elements in the project. Customer commitment is, thus, an important success factor.

#### *Decision Time*

In the words of Ken Auer, “Agile is about taking control of your own destiny to the point that you can” (eWorkshop, 2002). Successful agile software development teams are normally left on their own, thereby allowing them to take their own decisions and succeed (eWorkshop, 2002). Also, as close customer, business area and software developer collaborations are advocated by agile methodologies (Fowler and Highsmith, 2001), important project decisions are likely to be made within a short timeframe. Agile methodologies are perceived to likely succeed in environments where rapid communication is enabled (eWorkshop, 2002).

#### *Team Distribution*

One of the factors that is likely to positively influence the success of an agile software development project is the centralized organization of the teams. According to Ken Schwaber (eWorkshop, 2002), co-located teams are one of the important vehicles for successful communication, which is, in turn, identified by Scott Ambler (eWorkshop, 2002) as one of the important success factors of agile development. Companies involved in distributed international projects will be affected by the cultural, and political situations in those regions.

#### *Corporate Culture*

“To be agile is a cultural thing. If the culture is not right, then the organization cannot be agile” (Lindvall *et al.*, 2002). Having the right corporate culture is almost unanimously perceived by agile experts to be a necessary factor determining the introduction of agile methodologies (eWorkshop, 2002; Lindvall *et al.*, 2002). Since implementing agile methodologies require taking control of one’s own destiny to the maximum possible extent, the nature of organizations individuals work in is important. For example, agile is not appropriate in bureaucratic organizations (eWorkshop, 2002). A dynamic, and fast changing organization will find agile methods extremely suitable for it (Abrahamson *et al.*, 2002).

#### *Planning & Control*

One of the important aspects that characterize the implementation of agile software development methodologies is the nature of organizational, management, and project planning and control. For instance, documented plans, accompanied by quantitative performance measures are considered key to the

success of organizations practicing plan-driven methodologies. On the contrary, internalized plans, and qualitative control are considered to succeed organizations adopting agile practices (Boehm and Turner, 2003).

#### *Business/safety criticality*

The Part 3 of the *First eWorkshop on Agile Methods* (eWorkshop, 2002) had extensive discussions with the “agile experts” regarding how agile methodologies apply to systems where safety criticality, and reliability were major concerns. Barry Boehm believes that “agile methods fit applications that can be built quickly, and do not require extensive quality assurance” (eWorkshop, 2002). Boehm noted examples of failure of previous mission critical systems by deviating from the plan. Boehm’s statement was debated during the eWorkshop by Alistair Cockburn, who provided an example of a \$15M, 18-month, fixed-price, fixed-time project where agile methodologies were successful.

#### *Dynamism and Uncertainty*

The Agile Manifesto states that the proponents of agile software development value responding to change over following a plan (Fowler and Highsmith, 2001). Thus, agility, by definition, has a component of handling dynamism, and uncertainty built in it. Hakan Erdogmus and Joel Martin (eWorkshop, 2002) believe that agile methods are more appropriate for environments characterizing high degree of changes in requirements. However, they do not discard the value of agile methods for stable environments. According to Boehm and Turner (2003), agile methods are applicable in environments characterizing both high and low rates of change.

## 2.2. People Factors

The success of a software development project is often related to people factors. Human resources factors are also hypothesized as important factors for the success of agile software development projects.

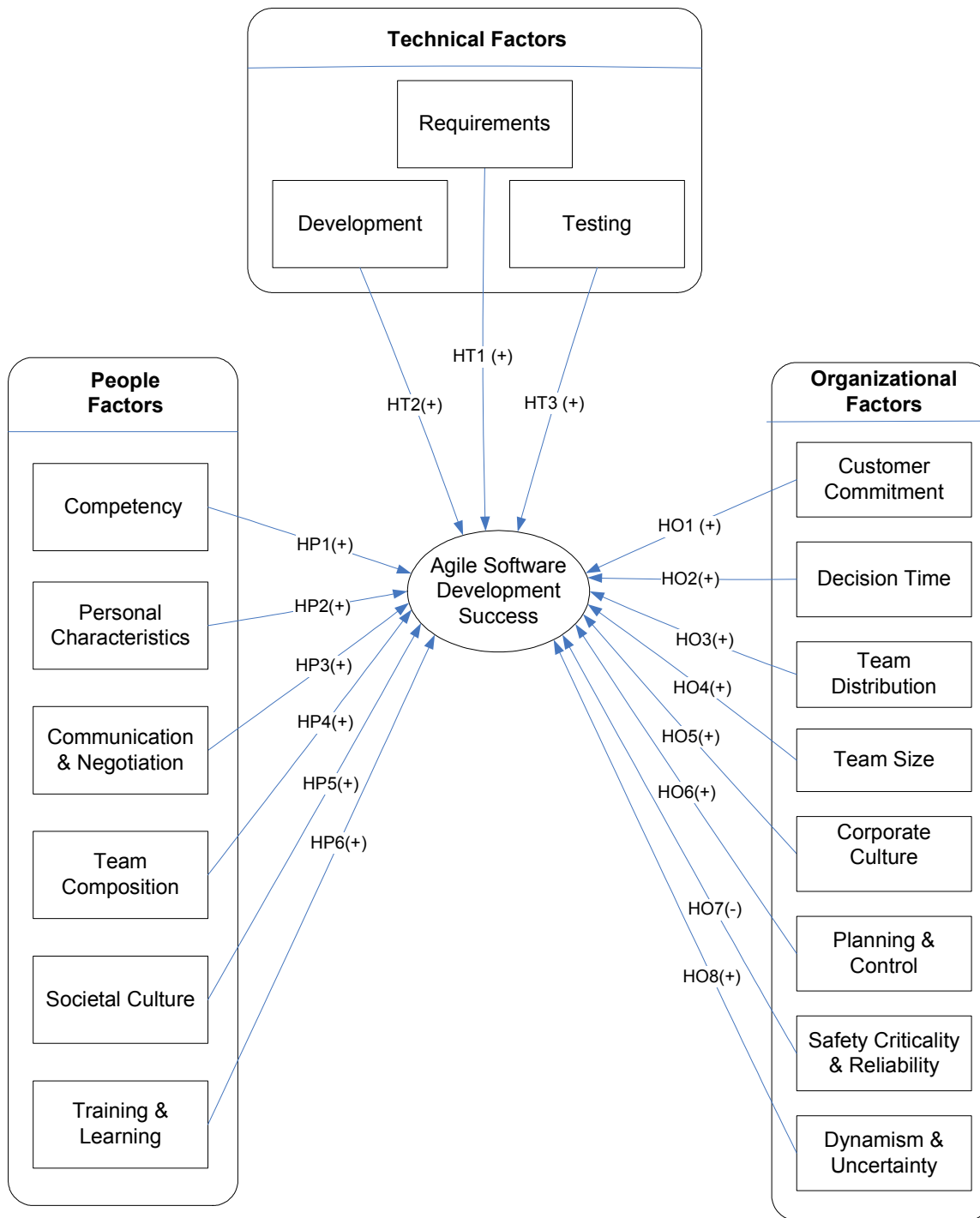
Some of the success factors mentioned below could equally be classified as organizational factors. However, instead of getting bogged down into the unimportant issue (according to the goals of this Thesis) of which category they belong to, we simply describe them here.

#### *Competency*

Competency means whether one has real-world experience in the technology domain, has built similar systems in the past, and possesses good interpersonal & communication skills (Lindvall *et al.*, 2002).

#### *Personal characteristics*

People factor plays a key role in the success of agile methods. According to Alistair Cockburn, “Good people are key to success with big teams”. Ken Auer believes that having high quality people does not necessarily mean having experienced ones. Team members may not be necessarily extremely experienced skilled people – having honesty, collaborative attitude, sense of responsibility, readiness to learn, and work with others are considered equally important, if not more. (eWorkshop, 2002).



**Figure 2.2: Theoretical Framework**

*Communication and Negotiation*

Communication plays an important role in the implementation of agile methodologies in a software development project (Turner and Boehm, 2003). Effective communication is identified as one of the important factors for agile methodologies to succeed (Ambler, 2005b). According to Scott Ambler

(eWorkshop, 2002), both processes and organization need to have in place a vehicle for communication. One of the important requirements for agility is fast and effective communication between developers, operations, support, customers, management, and business areas.

### *Team Composition*

Team composition is an important factor responsible for the success of agile methodologies. The number of experts in a team is considered important. What is important is the team composition with experts with actually building systems, rather than experience with agile methodologies. The composition of experts in a team is estimated (not firm) to be between 25-33%. Lower number of experts might also suffice in teams practicing pair programming (Lindvall *et al.*, 2002).

### *Societal Culture*

As any other human activity, software development is highly influenced by regional (local) culture. The Agile Manifesto (Fowler, 2002) lays emphasis on individuals and interactions between people over processes and tools. The agile principles (Fowler and Highsmith, 2001) require motivated individuals. People should be eager to learn from each other, be honest, collaborative, and responsible (eWorkshop, 2002). All these are affected by the societal cultural factors as well.

### *Training and Learning*

An important concern that has been raised in the previous literature is how much training is required for successfully implementing agile methodologies in an organization (Lindvall *et al.*, 2002). According to Bil Kleb, “‘continued learning’ is one of the fundamental differences between ‘agile’ and ‘others’” (eWorkshop, 2002). People should be eager to share information with one other, continuously learn. This increases the chances of agile practices.

## **3.3. Technical Factors**

### *Requirements*

Agile processes welcome changing requirements, even late in the development (Fowler and Highsmith, 2001). Unlike plan-driven methods, which are most useful in environments where there is low rate of change, agile methods have been successful in both high and low change environments. Whereas plan-driven methods work best with formalized project requirements, requirements capability, interface, quality, and predictable requirements, agile methods are successful even in environments where requirements undergo unforeseeable changes.

### *Development*

The success of plan-driven development is characterized by extensive design, longer increments in development. *Refactoring*, in which the internal structure of the existing code is changed without changing the external behavior of the system, is considered expensive in plan-driven methods. On the other hand, agile methodologies are considered to be successful in development environments characterizing simple design, short increments, and inexpensive refactoring (Boehm and Turner, 2003).

## Testing

Whereas documented test plans, and procedures characterize the success of plan-driven methods, executable test cases define the success of requirements and testing in agile software development (Boehm and Turner, 2003).

## 3. Conclusions

In this paper we attempted to review the important factors for the success of agile software development projects. We have also presented a conceptual framework illustrating the relationships between the different predictor variables and agile software development success.

Currently we are extending this work to uncover empirical evidence of factors affecting the success of agile software development projects, and to particularly identify (and possibly rank) them with respect to their criticality in succeeding such projects.

## References

P. Abrahamson *et al.* 2002. Agile software development methods – review and analysis. *VTT Publications 478*. <http://www.inf.vtt.fi/pdf/publications/2002/P478.pdf> (Last accessed: December 2005).

S.W. Ambler. 2002a. Lessons in Agility from Internet-Based Development. *IEEE Software*. March/April. pp. 66-73.

S.W. Ambler. 2002b. *Agile Modeling*. John Wiley & Sons.

S.W. Ambler. 2005a. Where is the Proof that Agile Methods Work. Online article that can be found at <http://www.agilemodeling.com/essays/proof.htm> (Last accessed: December 2005).

S.W. Ambler. 2005b. Communication on Agile Software Projects. Online article that can be found at <http://www.agilemodeling.com/essays/communication.htm> (Last accessed: December 2005).

S.W. Ambler. 2005c. Remaining Agile. Online article that can be found at <http://www.agilemodeling.com/essays/remainingAgile.htm> (Last accessed: December 2005).

B. Boehm and R. Turner. 2003. Observations on Balancing Discipline and Agility. *Proceedings of the Agile Development Conference (ADC'03)*. June. Salt Lake City, Utah, USA.

A. Cockburn. 2002. Learning From Agile Software Development – Part One. *CrossTalk: Journal of Defense Software Engineering*. October. <http://www.stsc.hill.af.mil/crosstalk/2002/10/cockburn.html>

A. Cockburn. 2002. Learning From Agile Software Development – Part Two. *CrossTalk: Journal of Defense Software Engineering*. November. pp. 9-12.

A. Cockburn. 2001. *Agile Software Development*. Addison-Wesley, 1<sup>st</sup> Edition.

M. Cohn and D. Ford. 2003. Introducing an Agile Process to an Organization. *IEEE Computer*. June. pp. 74-78.

A. Elssamadisy. 2001. XP on a Large Project – A Developer's View. <http://www.xpuniverse.com/2001/pdfs/EP202.pdf> (Last accessed: December 2005).

eWorkshop. 2002. Summary of the First eWorkshop on Agile Methods. <http://fc-md.umd.edu/projects/Agile/Summary/SummaryPF.htm>. April.

- M. Fowler and J. Highsmith. 2001. The Agile Manifesto. *Software Development Magazine*. August. <http://www.sdmagazine.com/documents/s=844/sdm0108a/0108a.htm> (Last accessed: December 2005).
- M. Fowler. 2002. The Agile Manifesto: Where It Came From and Where It May Go. <http://martinfowler.com/articles/agileStory.html> (Last accessed: December 2005).
- R. Gittins, S. Hope and I. Williams. 2001. Qualitative Studies of XP in a Medium Sized Business. *Proceedings of the 2<sup>nd</sup> International Conference on Extreme Programming and Flexible Processes in Software Engineering*. Sardina, Italy. May.
- J. Highsmith. 2000. Retiring Lifecycle Dinosaurs. *Software Testing and Quality Engineering Magazine*. <http://www.jimhighsmith.com/articles/Dinosaurs.pdf> (Last accessed: December 2005).
- E. R. Keith. 2002. Agile Software Development Processes: A Different Approach to Software Design. <http://www.agilealliance.com/articles/keithetteragileso/file> (Last accessed: December, 2005).
- N. Kini and S. Collins. 2003. Steering the Car: Lessons Learned from an Outsourced XP Project. <http://www.xpuniverse.com/2001/pdfs/XPU03.pdf> (Last accessed: December 2005).
- M. Lindvall *et al.* 2002. Empirical Findings in Agile Methods. *Proceedings of Extreme Programming and Agile Methods – XP/Agile Universe 2002*, pp. 197-207.
- Net Worth Consulting. 2004. *Time/Cost/Quality Trade-offs in the Budget Process*. Quantum<sup>2</sup> White Paper 10.25.04. Leadership Series: Measurement. [http://quantum.dialog.com/q2\\_resources/whitepapers/budget\\_process.pdf](http://quantum.dialog.com/q2_resources/whitepapers/budget_process.pdf) (Last accessed: December 2005).
- D. Opperthausen. 2003. Defect Management in an Agile Development Environment. *CrossTalk: The Journal of Defense Software Engineering*. September. <http://www.stsc.hill.af.mil/crosstalk/2003/09/0309Opperthausen.html> (Last accessed: December 2005).
- S.M. Pine. 2001. An Application of XP in a Multiple Team/Multi-Process Environment. <http://www.agilealliance.com/articles/pinestephenmanapplica/file> (Last accessed: December 2005).
- M. Poppendieck and T. Poppendieck. 2003. *Lean Software Development: An Agile Toolkit*. Addison-Wesley.
- PMI. 2004. *Project Management Body of Knowledge (PMBOK)*. 3<sup>rd</sup> Edition. Project Management Institute. USA.
- D.J. Reifer. 2002. How Good Are Agile Methods?. *IEEE Software*. July/August. pp. 14-17.
- B. Schatz, K. Schwaber and R.C. Martin. 2004. Best Practices in Scrum Project Management and XP Agile Software Development. White Paper. Object Mentor, Inc. <http://www.objectmentor.com/resources/articles/Primavera>. July.
- K. Schwaber and M. Beedle. 2002. *Agile Software Development with Scrum*. Prentice Hall, New Jersey, USA.
- K. Schwaber. 2004. *Agile Project Management with Scrum*. Microsoft Press, Redmond, Washington, USA.
- Shine Technologies. 2003. *Agile Methodologies: Survey Results*. Shine Technologies Pty Ltd. <http://www.shinetechnology.com> (Last accessed: December 2005).