Editor: Forrest Shull ■ Fraunhofer Center for Experimental Software Engineering, Maryland ■ fshull@fc-md.umd.edu

What Do We Know about Agile Software Development?

Tore Dybå and Torgeir Dingsøyr, *SINTEF*

Agile software development has had a huge impact on how software is developed worldwide. A 2005 survey of the US and Europe, for example, revealed that 14 percent of companies were using agile methods, and 49 percent of the companies aware of agile methods were interested in adopting them.[1]

We can view agile methods such as Extreme Programming (XP) and Scrum as a reaction to plan-

based or traditional methods, which emphasize a "rationalized, engineering-based approach,"[2] incorporating extensive planning, codified processes, and rigorous reuse.[3] In contrast, agile methods address the challenge of an unpredictable world, emphasizing the value competent people and their relationships bring to software development.[4] Table 1 summarizes these differences.

To clarify the effectiveness of agile methods, we reviewed the agile development literature and conducted a systematic study of what we know empirically about its benefits and limitations.

Agile Methods Overview

Three overviews from the first half of this decade describe the state of the art and practice of agile development in terms of lessons learned from applying various agile methods in industry.

The first is a literature review from a 2002 VTT Electronics technical report.[5] The report discussed the agile development concept in general, then presented experiences with 10 agile methods, and compared them with respect to the development phases they support and developer competence levels they require. The authors concluded that only the Dynamic Systems Development Method and the Rational Unified Process cover all development phases fully, while Scrum mainly covers aspects related to project management. They reported anecdotal evidence that agile methods are "effective and suitable for many situations and environments," but stated that few empirically validated studies support these claims. A follow-up analysis in 2003 stated that empirical support for the suggested methods remains scarce.[6]

In 2004, a review emphasizing agile development's history showed some of its roots in other disciplines.[7] In particular, it discussed relations between agile development and the Capability Maturity Model. The authors suggested that agile methods will eventually consolidate, just as object-oriented methods did. Furthermore, they believed that agile and traditional methods will have a symbiotic relationship, in which factors such as

Table 1

Traditional and agile perspectives on software development[4]

	Traditional view	Agile perspective
Design process	Deliberate and formal, linear sequence of steps, separate formulation and implementation, rule-driven	Emergent, iterative and exploratory, knowing and action inseparable, beyond formal rules
Goal	Optimization	Adaptation, flexibility, responsiveness
Problem-solving process	Selection of the best means to accomplish a given end through well-planned, formalized activities	Learning through experimentation and introspection, constantly reframing the problem and its solution
View of the environment	Stable, predictable	Turbulent, difficult to predict
Type of learning	Single-loop/adaptive	Double-loop/generative
Key characteristics	Control and direction Avoids conflict Formalizes innovation Manager is controller Design precedes implementation	Collaboration and communication; integrates different worldviews Embraces conflict and dialectics Encourages exploration and creativity; opportunistic Manager is facilitator Design and implementation are inseparable and evolve iteratively
Rationality	Technical/functional	Substantial
Theoretical and/or philosophical roots	Logical positivism, scientific method	Action learning, John Dewey's pragmatism, phenomenology

the number of people working on a project and the application domain, criticality, and innovativeness will determine which process to select.

In 2005, a study looked at the state of research on XP, agile software development, and agile modeling.[8] With respect to XP, the authors reported a small number of case studies and experience reports promoting XP's success. A more well-established stream of research supports pair programming, and some studies also support iterative development. The authors recommended the separate study of other core XP practices to identify which ones work. Furthermore, they saw challenges in matching agile software development methods with standards such as ISO, and they argued that this area also needs further research.

What We Know on the Basis of Empirical Studies

In 2008, we undertook a review of agile software development, the first—and so far the only—review to systematically evaluate, synthesize, and present the available empirical findings.[9] We aimed to determine what's known about agile development's benefits and limitations, the strength of the evidence supporting these findings, and the implications for the software industry and research community.

Our review identified 1,996 studies from literature searches, 36 of which we found to be research studies of acceptable rigor, credibility, and relevance to include in the review. Of the 36, 33 were primary studies and three were secondary studies. Most of the studies were published in 2004 and 2005 (see Figure 1). The studies investigated XP almost exclusively (25 out of 36), and only a few of the XP studies addressed mature development teams. A clear finding of the review is that we need to increase both the number and quality of agile studies. In particular, agile project management methods such as Scrum, which are popular in industry, warrant further attention.

More detail on the studies we selected is available in a Web appendix to this article at www.computer.org/software/webextra.html.

The studies fell into four thematic groups: introduction and adoption, human and social factors, perceptions of agile methods, and comparative studies. We identified several reported benefits and limitations of agile development in each of these themes and summarize the results here.

Introduction and Adoption

These studies didn't provide a unified view of current practice. Instead, they offered a broad picture of experience and some contradictory findings.

XP seemed difficult to introduce in large, complex organizations but easier in other organization types. Most studies reported that agile development practices are easy to adopt and work well. The benefits appeared in customer collaboration, work processes for handling defects, learning among developers, thinking ahead for management, focusing on current work for engineers, and software estimation.

Lean development didn't work well for one team that tried it. Some studies saw pair programming as inefficient, and some studies claimed that XP works best with experienced development teams. One study reported the further limitation, which the literature repeatedly mentions, of lack of attention to design and architectural issues.

Human and Social Factors

A recurring study theme was human and social factors and how they affect, and are affected by, agile development methods. XP thrived in radically different environments—from organizations having a hierarchical structure to those having

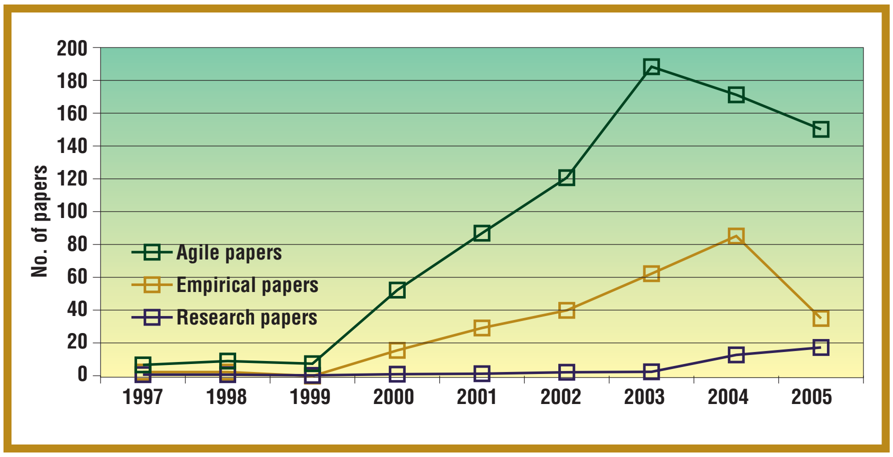

Figure 1. The number of papers identified in the systematic review of agile development studies by year. The figure separates the papers into three groups: all agile papers, the proportion of the agile papers that were empirical, and the proportion of the empirical papers that were research papers.

little or no central control. It seems possible to adopt XP in various organizational settings.

Furthermore, researchers have studied conversation, standardization, and progress tracking and described them as mechanisms for creating awareness in teams and organizations. Good interpersonal skills and trust were important characteristics for a successful XP team.

Perceptions of Agile Methods

Many studies sought to identify how different groups perceive agile methods.

Studies of customer perceptions reported customer satisfaction with opportunities to get and give feedback. However, the on-site customer's role can be stressful and unsustainable for long periods.

Developers were mostly satisfied with agile methods. Companies that use XP have reported that employees are more satisfied with their jobs and with the product. The findings regarding pair programming's effectiveness were mixed. Several developers regarded it as an exhausting practice because it requires heavy concentration.

University students perceived agile methods as an opportunity for relevant training for future work and believe that these methods improve team productivity. However, they reported that pair programming was difficult when the skill differences between the pair members were large. In addition, many students reported that test-first development was difficult.

Comparative Studies

These studies compared variations of traditional development to variations of agile development. They showed that traditional and agile development methods use different project management practices.

Some studies suggested that agile projects can incorporate changes more easily and demonstrate business value more efficiently than traditional projects. In addition, it seems that agile project management can be combined with overall traditional principles, such as the stage-gate project management model. One limitation that came up was that team members are less interchangeable in agile teams, which has consequences for how projects are managed.

With respect to the productivity of agile and traditional teams, three of the four comparative studies that addressed this issue found that using XP increases productivity in terms of lines of code per hour. However, none of these studies had an appropriate recruitment strategy to ensure an unbiased comparison. Findings from several of the noncomparative studies indicated that the subjects themselves believe agile methods increase productivity.

Practical Implications

Our review shows that there have been many promising studies of the use of agile methods. Although the studies identified serious limitations, such as the unsustainability of the on-site customer's role for long periods and the difficulty of introducing agile methods into large and complex projects, the review results suggest that agile methods can improve job satisfaction, productivity, and customer satisfaction.

The strongest, and probably most relevant, evidence for practice is from the studies of mature agile teams, which suggests that focusing on human and social factors is necessary to succeed. Specifically, it seems that a high level of individual autonomy must be balanced with a high level of team autonomy and corporate responsibility. It also seems important to staff agile teams with people who have faith in their own abilities combined with good interpersonal skills and trust.

Evidence also suggests that instead of abandoning traditional project management principles, organizations should exploit these principles, such as stage-gate project management models, and combine them with agile project management. The evidence also suggests that agile methods aren't necessarily the best choice for large projects. So, we recommend that practitioners carefully study their projects' characteristics and compare them with the relevant agile methods' required characteristics.

Our review clearly shows the need for more and better research to determine the situations in which practitioner advice on agile development can be suitably applied. We urge companies to participate in research projects that target goals relevant for the software industry.[10] Action research tries to provide practical value to client organizations while simultaneously contributing to new theoretical knowledge.[11] It offers one way to organize collaboration between industry and researchers that would be highly relevant for a nascent field such as agile software development.

References

1. "Software and Services in Large Enterprises," *Business Technographics*, Forrester Research, 2005; www.forrester.com/Research/Document/Excerpt/0,7211,38659,00.html.
2. T. Dybå, "Improvisation in Small Software Organizations," *IEEE Software*, vol. 17, no. 5, 2000, pp. 82–87.

3. B. Boehm, "Get Ready for Agile Methods, with Care," *Computer*, vol. 35, no. 1, 2002, pp. 64–69.
4. S. Nerur and V. Balijepally, "Theoretical Reflections on Agile Development Methodologies," *Comm. ACM*, vol. 50, no. 3, 2007, pp. 79–83.
5. P. Abrahamsson et al., *Agile Software Development Methods: Review and Analysis*, tech. report, VTT Electronics, 2002.
6. P. Abrahamsson et al., "New Directions on Agile Methods: A Comparative Analysis," *Proc. 25th Int'l Conf. Software Eng.* (ICSE 03), IEEE CS Press, 2003, pp. 244–254.
7. D. Cohen, M. Lindvall, and P. Costa, "An Introduction to Agile Methods," *Advances in Computers, Vol. 62: Advances in Software Engineering*, M.V. Zelkowitz, ed., Elsevier, 2004, pp. 1–66.
8. J. Erickson, K. Lyytinen, and K. Siau, "Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research," *J. Database Management*, vol. 16, no. 4, 2005, pp. 88–100.
9. T. Dybå and T. Dingsøyr, "Empirical Studies of Agile Software Development: A Systematic Review," *Information and Software Technology*, vol. 50, nos. 9–10, 2008, pp. 833–859.
10. T. Dybå, B. Kitchenham, and M. Jørgensen, "Evidence-Based Software Engineering for Practitioners," *IEEE Software*, vol. 22, no.1, 2005, pp. 58–65.
11. D. Avison et al., "Action Research," *Comm. ACM*, vol. 42, no. 1, 1999, pp. 94–97.

Tore Dybå is chief scientist and a research manager at SINTEF Information and Communication Technology. Contact him at tored@sintef.no.

Torgeir Dingsøyr is a senior scientist at SINTEF Information and Communication Technology and an adjunct associate professor in the Department of Computer and Information Science, Norwegian University of Science and Technology. Contact him at torgeird@sintef.no.

IEEE Software

STAFF

Senior Lead Editor
Dale C. Strok
dstrok@computer.org

Content Editor
Brian Brannon

Magazine Editorial Manager
Jenny Stout

Staff Editors
Dennis Taylor and Linda World

Publications Coordinator
software@computer.org

Production Editor/Webmaster
Jennie Zhu

Contributors
Molly Gamborg, Rita Scanlan, Joan Taylor, and Alex Torres

Director, Products & Services
Evan Butterfield

Senior Editorial Services Manager
Crystal R. Shif

Senior Business Development Manager
Sandra Brown

Digital Library Marketing Manager
Georgann Carter

Senior Advertising Coordinator
Marian Anderson
manderson@computer.org