The Journal of Systems and Software 85 (2012) 1213–1221

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

A decade of agile methodologies: Towards explaining agile software development

ARTICLE INFO

Keywords:
Agile software development
Theory
Software engineering
Information systems
eXtreme programming, XP
Scrum
Lean software development
Crystal method
Feature-driven development

ABSTRACT

Ever since the agile manifesto was created in 2001, the research community has devoted a great deal of attention to agile software development. This article examines publications and citations to illustrate how the research on agile has progressed in the 10 years following the articulation of the manifesto. Specifically, we delineate the conceptual structure underlying agile scholarship by performing an analysis of authors who have made notable contributions to the field. Further, we summarize prior research and introduce contributions in this special issue on agile software development. We conclude by discussing directions for future research and urging agile researchers to embrace a theory-based approach in their scholarship.

1. Introduction

The articulation of the agile manifesto in 2001[1] – a little over a decade ago – has brought unprecedented changes to the software engineering field. Indeed, the transformation that the manifesto has brought in its wake is quite remarkable. It is hard to think of a decade in the twentieth century that has witnessed the introduction of so many software methods, tools, techniques, and best practices. While this unparalleled growth has been readily accepted by many practitioners, much work has still to be undertaken to bring coherence to the current discourse on agility.

As with any nascent discipline, the early years of agile development were marked by exuberance of a few and by scepticism among many. A host of methods, adhering to varying degrees to the tenets of the manifesto, appeared on the landscape. These include eXtreme programming (XP), scrum, lean software development, feature-driven development (FDD), and crystal methodologies, to name but a few. Broadly speaking, all these methods endeavoured to address the core principles of the manifesto. First, there was a distinct move towards collaborative development, with people being accorded privileges over processes that formerly constrained them. Second, a dominant "lean" mentality was advocated with a view to minimizing unnecessary work, particularly with regard to the creation of wasteful documentation. While this was misconstrued by many to mean "no documentation", the discerning realized that this meant documenting only what was absolutely necessary and nothing more. Third, customers/stakeholders were no longer just at the fringes of software development, but actively shaped and guided the evolution of the end software product or service. Fourth, there was an acceptance of the fact that uncertainty was a part and parcel of software development, and that the inherent tendency to control variations through statistical and other means was futile.

After much discussion about the idiosyncrasies of the many methods that were proposed, the conversation shifted to the relative merits of plan-driven and agile methods, the need to have a balanced approach, the circumstances under which each would be more appropriate, and so forth (for example, see Boehm and Turner, 2004). In recent times, the attention has been focused on issues related to managing the actual project—agile planning, control, and estimation, streamlining flow of stories (e.g., Kanban), using lean six-sigma, and so forth. Most of these ideas have spawned a number of practices that are claimed to be efficacious, but empirical validation of such assertions is lacking.

The early research on agile focused, quite understandably, on issues related to the adoption of agile methods (e.g., Boehm, 2002; Nerur et al., 2005) and on the efficacy of pairs vis-à-vis individuals in software development (Nawrocki and Wojciechowski, 2001; Williams et al., 2000). Other studies have investigated various aspects of team dynamics -e.g., trust, self-organization, and communication) (Moe et al., 2009), consequences of test-driven development (Erdogmus et al., 2005; Janzen and Saiedian, 2005), adoption and post-adoption issues (Cao et al., 2009; Mangalaraj et al., 2009), challenges of implementing agile in distributed settings (Ramesh et al., 2006), and the like. Despite the copious research on agile software development and its ramifications, one cannot help but sense a lack of a unified framework that brings coherence to the seemingly disparate streams of research being pursued. Clearly, more work has to be done to articulate quintessential principles of agile software development that are at once unequivocal and useful for practice. The goal of this special issue is to draw attention to this imperative and to present articles that could further our understanding of the myriad implications of agile software development.

The rest of the article is structured as follows. In the next section we present an overview of research on agile software development. Specifically, we examine publications and citations related to agile development to delineate the structure of the field. Subsequently, we summarize prior research on agile, followed by a brief account

[1] See http://agilemanifesto.org/.

0164-1212
doi:10.1016/j.jss.2012.02.033

of the contributions made by the papers in this special issue. Finally, the conclusions and directions for future research are discussed.

2. An overview of research on agile software development

2.1. Agile principles and agility

According to the agile principles enunciated in the agile manifesto1, motivated and empowered software developers – relying on technical excellence and simple designs – create business value by delivering working software to users at regular short intervals. These principles have spawned a number of practices that are believed to deliver greater value to customers. At the core of these practices is the idea of self-organizing teams whose members are not only collocated but also work at a pace that sustains their creativity and productivity. The principles encourage practices that accommodate change in requirements at any stage of the development process. Furthermore, customers (or their surrogates) are actively involved in the development process, facilitating feedback and reflection that can lead to more satisfying outcomes. The principles are not a formal definition of agility, but are rather guidelines for delivering high-quality software in an agile manner. While individual principles and practices of agile development were not entirely new to the software community, the way in which they were put together into a cogent "theoretical and practical framework" was certainly novel (Williams and Cockburn, 2003). Ever since the manifesto was articulated, practitioners and researchers have been trying to explicate agility and its different facets. At its core, agility entails ability to rapidly and flexibly create and respond to change in the business and technical domains (Henderson-Sellers and Serour, 2005; Highsmith and Cockburn, 2001). Other aspects of agility explored include lightness or leanness (i.e., having minimal formal processes) (Cockburn, 2007) and related concepts such as nimbleness, quickness, dexterity, suppleness or alertness (Erickson et al., 2005). In essence, these ideas suggest a "light' methodology that promotes manoeuvrability and speed of response" (Cockburn, 2007).

More formal definitions of agility have started to appear in the recent past, drawn mainly from manufacturing and management domains, where agile appears to have its roots. For Henderson-Sellers and Serour (2005), agility involves both the ability to adapt to different changes and to refine and fine-tune development processes as needed. Lee and Xia (2010) define software development agility "as the software team's capability to efficiently and effectively respond to and incorporate user requirement changes during the project life cycle." Conboy (2009) provides by far the most comprehensive definition of software development agility by systematically examining its various facets and definitions from related disciplines. He makes a distinction between agility, flexibility, and leanness—in fact, agility is conceptualized to include and go beyond both flexibility and leanness. While flexibility relates to the ability of a systems development method to "create change, or proactively, reactively, or inherently embrace change in a timely manner, through its internal components and its relationships with its environment", leanness captures the "contribution to perceived customer value through economy, quality, and simplicity." Thus, Conboy (2009, p. 340) defines software development agility as the continued readiness "to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment."

While leanness emphasizes cost reduction through eliminating waste and inefficiencies, agility treats leanness – i.e., cost reduction through waste elimination – as a qualifier to focus more heavily on creating effective responses and valuable outcomes (Agarwal et al., 2006). Thus, leanness may be perceived as efficiency oriented, while agility entails embracing lean processes with an emphasis on realizing effective outcomes. Lyytinen and Rose (2006) suggest that agility is achieved through learning processes involving both exploration and exploitation. The next section highlights the extent of research on agile development undertaken during the past decade across countries and across journals/conferences.

2.2. Research on agile software development

A literature search in the ISI Web of Science[2] identified 1551 research papers on agile software development that were published between 2001 and 2010, inclusive. As shown in Fig. 1, the number of journal articles as well as conference papers has been steadily increasing until 2010. A plausible explanation for the decline in the number of conference publications in 2010 is that the 2010 Agile conference was not indexed in ISI Web of Science database. The increase in journal articles indicates that the research field is maturing. A systematic review of empirical research published before 2005 revealed a lack of theoretical and methodological rigor (Dybå and Dingsøyr, 2008). The total number of publications shows that agile development has received much interest from the academic community; however, most of the research is inspired by practices emerging in industry.

Our literature search also permitted us to examine the extent of agile research undertaken in different countries. Fig. 2 shows the number of publications by country, darker colours indicating a larger volume of agile papers. Note that although the majority of the articles originate in the US, Canada and Western Europe, agile software development has been a research theme on all continents, in a total of 63 countries (see Appendix 1 for details).

We also tried to identify the popular conferences and journals in which publications on agile research appear. It can be seen from Fig. 3 that the *International Conference on Agile Software Development* ("XP") based in Europe has been the main forum for agile research, followed by *Agile*[3] in the US. This is not surprising, as the two conferences focus exclusively on issues related to agile software development. Other popular avenues for agile research are *Profes* and *EuroSPI* – both of which focus on process improvement – and the *International Conference on Software Engineering* (ICSE). *EuroMicro* and the *Conference on Software Engineering Education and Training* (CSEE&T) have also been able to attract some papers on agile. Finally, the IFIP community and the *International Conference of Global Software Engineering* (ICGSE), as well as the *International Symposium on Empirical Software Engineering and Measurement* (ESEM), have devoted some attention to this topic. Thus, several research communities have focused on agile development. A vast majority of the papers (1064 out of 1302) are from the top ten conferences. The remaining 238 articles are spread over about 200 other forums. This shows that agile development has received widespread attention across various scientific communities.

An overview of journal publications (see Fig. 4) reveals that *IEEE Software* has the largest number of papers, followed by the *Journal of Systems and Software*, *Information and Software Technology*, and *Empirical Software Engineering*. The leading publisher of agile

[2] We used the following search: Topic=("Agile development" OR "Agile software development" OR "Agile Methodologies" OR Ägile methods" OR "Agile Project Management" OR "Lean development" OR "Lean software development" OR "Scrum" OR "Extreme Programming" OR "Pair Programming" OR "Test-Driven Development") Refined by: Subject Areas=(COMPUTER SCIENCE OR ENGINEERING OR TELECOMMUNICATIONS OR OPERATIONS RESEARCH MANAGEMENT SCIENCE) AND Document Type=(PROCEEDINGS PAPER OR REVIEW OR ARTICLE OR BOOK CHAPTER).

[3] These numbers also include the previous Agile Development Conference and the XP Agile Universe, but not all years have had the proceedings indexed.

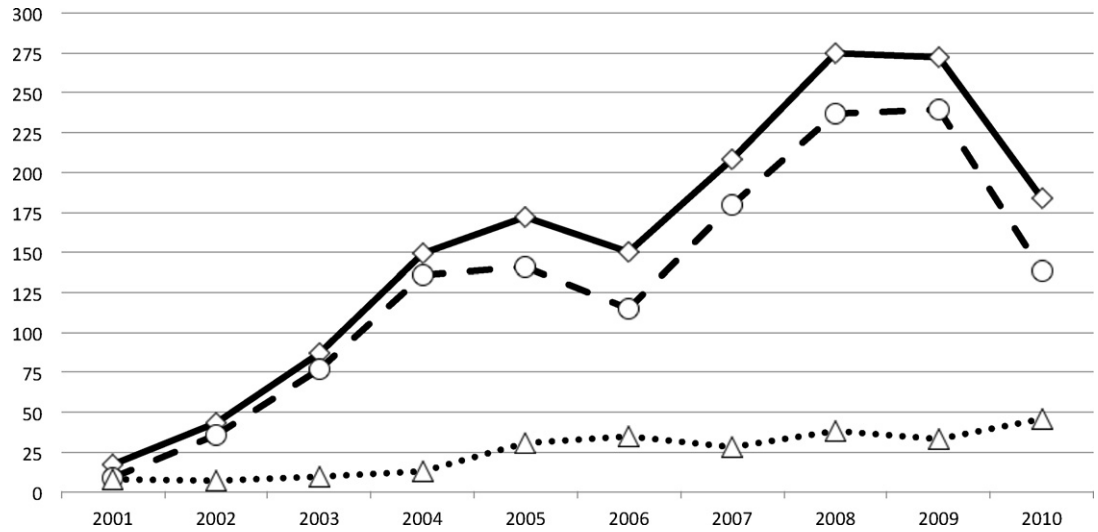

Fig. 1. Publications on agile software development from 2001 to 2010, total number (top), conference papers (middle) and journal articles (bottom).

Fig. 2. Publications on agile software development by country. Darker colour indicates more publications. See Appendix 1 for details.

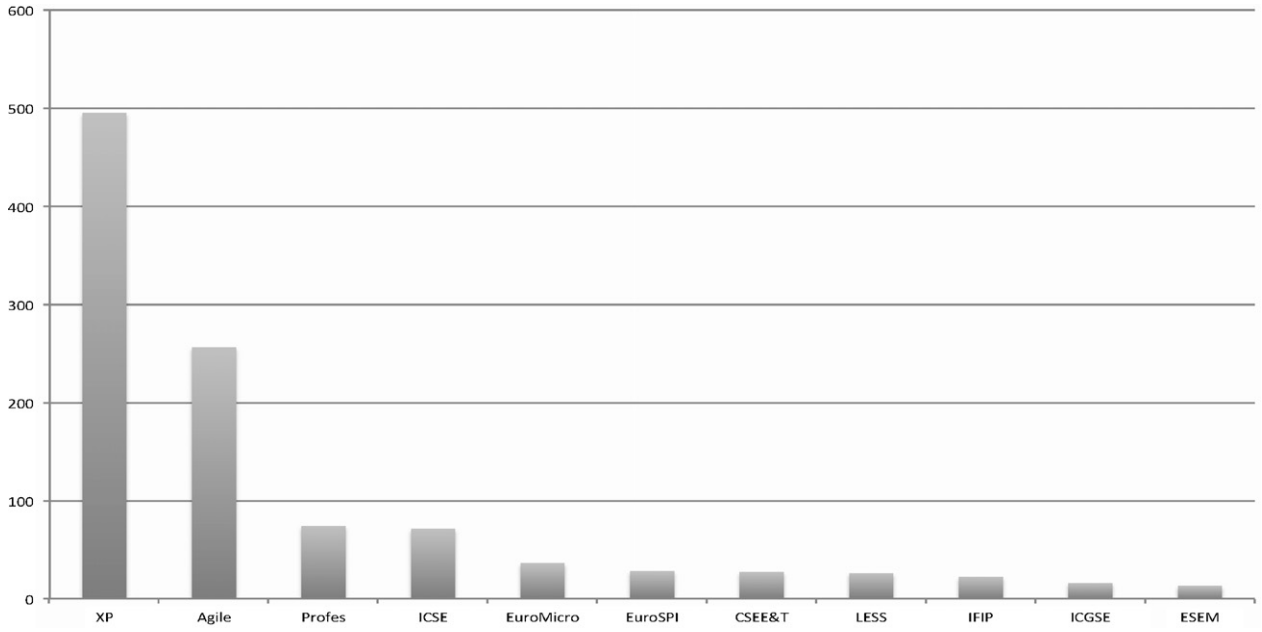

Fig. 3. Number of papers by conference.

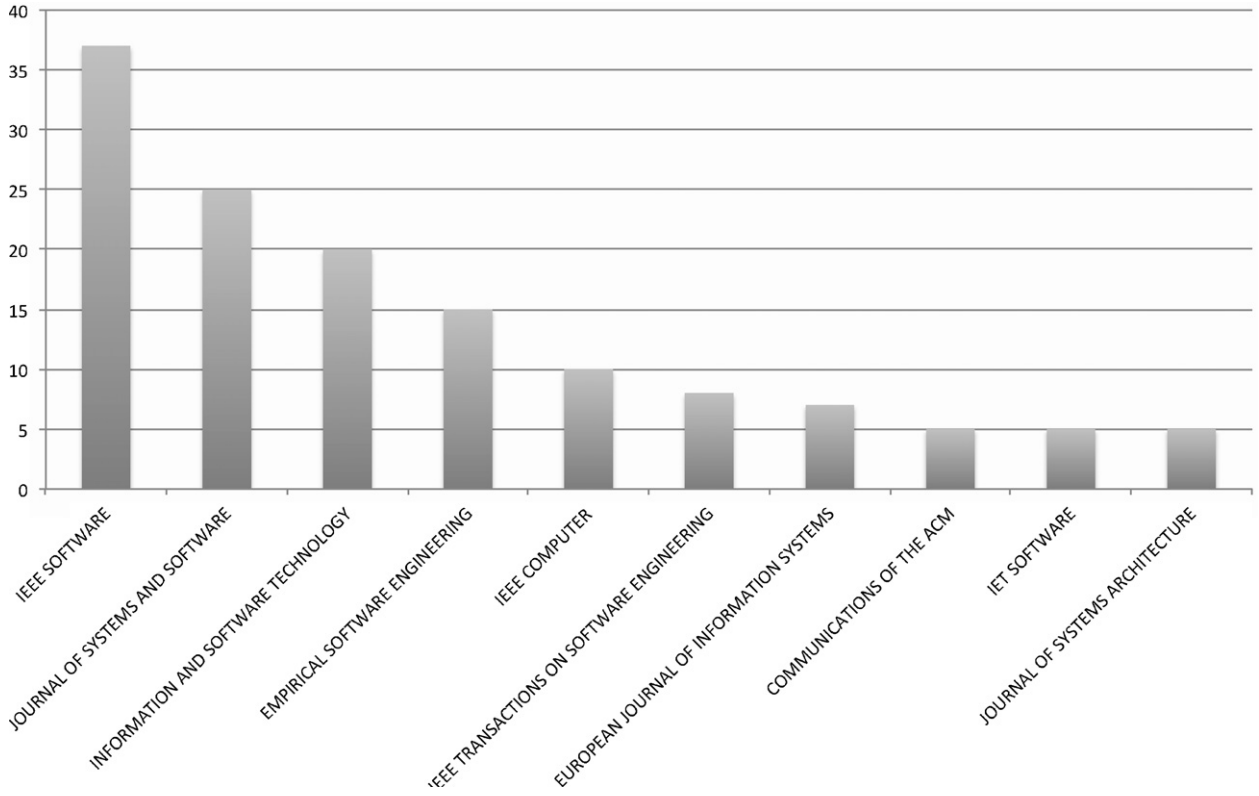

Fig. 4. Number of papers in scientific journals.

articles among non-software engineering journals is the *European Journal of Information Systems*, thanks to a special issue on the topic. Yet another popular outlet, *Communications of the ACM*, has published five articles on agile software development. Thus, we see that the topic has gained traction not just in software engineering but in other areas as well.

2.3. Seminal contributors and their relationships

There is perhaps no better way to understand a field than to identify its seminal sources of information and the relationships among them. Indeed, experts in bibliometric studies, such as McCain (1990), White and Griffith (1981), and White (1990), have elaborated on the use of co-citation analysis to delineate the conceptual underpinnings of disciplines. Specifically, researchers have used either authors (e.g., Nerur et al., 2008) or documents (e.g., Ramos-Rodríguez and Ruíz-Navarro, 2004) as the units of analysis. Whatever be the case, the unit of analysis – author or the article written by the author – is regarded as a concept (or concepts) that it promulgates. Co-citation analysis, whether it is based on co-citations of authors or documents, rests on the premise that joint citations between two units occur when they share some conceptual similarity. This study used author co-citation analysis (ACA) to unravel dominant conceptual themes in the agile literature.

The procedure followed is very consistent with those outlined by McCain (1990) and Nerur et al. (2008). The first step was to identify the authors most frequently cited in the agile literature. ISI's Web of Science was used for this purpose. Our search[4] yielded 452 articles. The cited references of each of these articles were then used to get a list of the most frequently cited authors. Fifty-one authors were included in our final analysis. A 51×51 matrix of raw co-citation frequencies was then computed based on a "cited references search" involving each of the 51 authors. Specifically, the co-citation frequency between a pair of authors was obtained by determining the number of matching records in their respective cited references. The input to the cluster analysis program was a correlation matrix derived from the 51×51 matrix of raw co-citation counts. Consistent with other ACA studies, we used the Ward's method. The results are shown in Fig. 5.

Fig. 5 shows the key conceptual themes that have appeared in the agile literature. Distances in the figure provide a sense of the level of conceptual similarity between two authors, with shorter distances implying greater thematic closeness in their writings. For example, the short linkage distance between Fowler and Gamma – both of whom have written about patterns – implies that their works are highly related. As mentioned earlier, the formative years of the field saw the proliferation of research related to the differences between process-oriented approaches such as CMM/CMMI and agile methods such as XP. Further, there were extensive debates on topics related to reconciling differences between agile and erstwhile practices, striking a balance between traditional and agile, using a risk-driven approach to choosing practices, and so forth. The presence of pair-programming and test-driven development is no surprise, since a lot of research has been devoted to these practices. Sharp and Robinson distinguish themselves by their work on XP, particularly in the context of organizational culture, distributed cognition, and the role of physical artefacts in agile development (see Sharp and Robinson, 2006, 2008; Sharp et al., 2009). The influence of patterns – both analysis and design patterns – based on the works of Fowler and Gamma is also evident in the analysis. Robert Martin's book on agile software development shows the role of patterns in agile development; hence the link among Martin, Fowler, and Gamma. The next section identifies the various theoretical perspectives used in prior agile research.

[4] Same search as in endnote 2, but limited to articles or reviews.

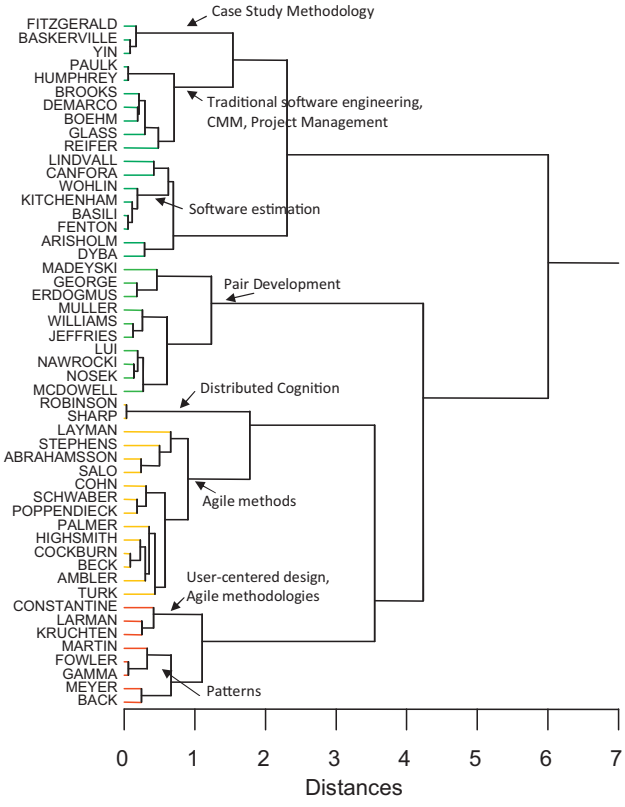

Fig. 5. Key research themes in agile software development. Cluster analysis of seminal authors using the Ward's method.

2.4. Theoretical exploration of agile development

Agile development evolved from the personal experiences and collective wisdom of the consultants and thought leaders of the software community. While most individual agile practices have intuitive appeal – as they are based on generally accepted management principles – they certainly lacked theoretical underpinnings or empirical support for their stated benefits, when initially conceived. Thus, there was a pressing need for theoretical perspectives that throw light on how these disparate practices and their interactions could produce valued outcomes. Theoretically comprehending the distinction between agile methods and their traditional counterparts was another concern begging for research attention. Some early studies sought to address these concerns. For instance, Nerur and Balijepally (2007) illustrate that the shifts in the approach to software development reflected in the agile methods has parallels to similar shifts in design thinking evident in several disparate fields, e.g., architecture and strategic management domains. They suggested the theory of holographic organization and its various principles as a theoretical lens to explore agile development.

To get a sense of the various theoretical perspectives used in studies on agile development during the last decade, we did a quick analysis of the topic search of the 452 articles published between 2001 and 2010 that we identified earlier for author co-citation analysis. Using ISI's Web of Science we searched through the topics of these 452 articles using keywords outlined in Table 1. The various theoretical perspectives and the number of articles identified by ISI's Web of Science using these terms in the topic results are showcased in Table 1. An obvious limitation of the approach used for identifying these articles relates to the keywords used (which are based on our understanding of the popular theoretical perspectives found in agile studies) and the search procedure adopted here (i.e., searching in the topic field of ISI Web of Science database). However, we believe this provides a broad sense of the various theoretical perspectives used in agile studies and their relative popularity.

Table 1
Theoretical perspectives used in agile research.

Theoretical Perspective	Number of Articles	Article(s)
Knowledge management	9	Dingsøyr and Hanssen (2002) Holz and Maurer (2002) Sena and Shan (2002) Doran (2004) Fang et al. (2004) Bellini et al. (2005) Crawford et al. (2006) Salazar-Torres et al. (2008) Chan and Thong (2009)
Personality	6	Sfetsos et al. (2006) Choi et al. (2008) Layman et al. (2008) Sfetsos et al. (2009) Acuna et al. (2009) Hannay et al. (2010)
Organizational learning	1	Holz and Maurer (2002)
Double loop learning	1	McAvoy and Butler (2007)
Triple-loop learning	1	McAvoy and Butler (2007)
Complex adaptive systems	2	Meso and Jain (2006) Socha and Walter (2006)
Social facilitation	2	Arisholm et al. (2007) Balijepally et al. (2009)
Adaptive Structuration theory	1	Cao et al. (2009)
Chaos theory	1	Levardy and Browning (2009)
Complexity theory	1	Falessi et al. (2010)
Coordination theory	1	Pikkarainen et al. (2008)
Distributed cognition	1	Sharp and Robinson (2008)
Evolutionary theory of knowledge	1	Northover et al. (2006)
Fuzzy set theory	1	Mafakheri et al. (2008)
Game theory	1	Hazzan and Dubinsky (2005)
Graph theory	1	Zimmer (2003)
Socio technical	1	Johannessen and Ellingsen (2009)
Teamwork model	1	Moe et al. (2010)
Theory of diagnosis	1	Trinidad et al. (2008)

As is evident from Table 1, knowledge management, personality, and organizational learning and related perspectives have been more popular with agile researchers. As software development is a knowledge creation activity, knowledge management should be an attractive perspective when exploring knowledge generation in software teams in general and agile teams in particular. Similarly, personality theories (e.g., Big Five personality theory) should be useful in exploring the interpersonal dynamics of collocated agile teams and programming pairs. As agile principles of change readiness and adaptability are expected to foster a learning environment in agile teams, organizational learning and related perspectives would be a logical choice for researchers when exploring learning outcomes of agile development. As is also apparent from Table 1, other theoretical perspectives have been used to a much lesser extent. Most importantly, a majority of agile studies do not seem to be concerned about any theoretical underpinnings for their research exploration, which reinforces the general popular perception that agile research tends to be a-theoretical.

3. Towards a theory of agile software development

In this section, we first describe the state-of-the art in agile development, and then proceed to place this special issue in context.

3.1. An overview of prior research

Introductions to and overviews of agile development are provided by Abrahamsson et al. (2002), Cohen et al. (2004), Erickson et al. (2005), and Dybå and Dingsøyr (2008). These four reports describe the state-of-the-art and state-of-the-practice in terms of characteristics of the various agile methods and lessons learned from applying such methods in industry. In addition, the book, *Agile Software Development: Current Research and Future Directions* (Dingsøyr et al., 2010), contains eleven overviews of the main streams within agile research, structured in chapters explaining foundations and background of agile development, agile methods in practice, and principal challenges and new frontiers.

From 2003 until 2011 five special issues and one special section on agile software development have been published, including 32 articles. The most common agile methods described were XP and Scrum. An examination of these special issues revealed that most articles were devoted to furthering our understanding of agile concepts. Other dominant topics included adoption and/or adaptation of agile, reconciliation of the tension between agile and plan-driven development (i.e., flexibility and control), and evaluation of adoption issues in environments that are not inherently conducive to agile. We will now give a short summary of these special issues and the special section:

In 2003, Williams and Cockburn (2003) edited a special issue in *Computer* titled, "Agile Software Development: It's about Feedback and Change". The primary emphasis of the special issue was on determining how to blend agile methodologies with plan-driven approaches to software development. The six articles included covered the history of iterative and incremental development, the debate on mixing agile and plan-driven development, and how and when to mix these two approaches. Furthermore, the special issue reported experience on the use of XP and Scrum, as well as on introducing agile processes into an organization working in an ISO 9000 or CMMI environment.

The second special issue appeared in the *Journal of Database Management in 2005* (Siau, 2005). In addition to a review of the state of research on XP and agile methods, the issue covered topics related to adoption of agile methods, process improvement, XP, and the underlying assumptions of agile.

In 2009, Abrahamsson et al. (2009) selected seven articles for a special issue in the *European Journal of Information Systems* to further our understanding of various phenomena in agile system development. The title of the special issue editorial was "'Lots done, more to do': the current state of agile systems development research". The papers not only addressed the fundamental question of what constitutes 'agility' and agile methods, but also demonstrated approaches to broadening the scope of the applicability of agile concepts.

Ågerfalk, Fitzgerald, and Slaughter also edited a special issue in 2009 (Ågerfalk et al., 2009). Seven papers were published in the *Information Systems Research* under the banner, "Flexible and Distributed IS Development: State of the Art and Research Challenges". The papers attempted to explore and/or define the central concept of agility, the enablers and inhibitors of agility, the question of how to balance flexibility and control, the circumstances under which agile methods are most effective, and the challenges of agile in distributed projects.

Yet another special issue was published in *Software Practice and Experience* 2011, which was edited by Greer and Hamon (2011). The papers addressed a range of research areas including the application of agile methods to safety critical software development, the relationship between agile development and user experience design, and the measurement of flow in lean software development.

Finally, in 2011, Dybå (2011) edited a special section in the *Journal of Information and Software Technology*, based on best papers from the XP2010 conference. The four articles that were published described the relationship between organizational culture and development of agile methods, customer involvement in agile projects, self-management, and the evolution of the practice of agile information systems development within a company over a 10-year period.

3.2. This special issue

For this special issue, we asked for contributions that critically reflect on the current status of research and practice in agile development. In particular, we were looking for contributions questioning and exploring the theoretical underpinnings of agile and lean development and the agile manifesto. We received a total of 21 submissions, of which five were selected for the special issue. Three of these articles focus on specific aspects of agile practices – coordination, decision making and post adaptive use – while the last two articles provide insight on broad topics of agile development – a grounded theory of software development, and a review of experience reports on "lean and agile" software development. We describe these contributions in more detail below.

In their article, "Coordination in co-located agile software development projects", Strode, Huff and Hope link agile development to theory of coordination, using a model with three components: synchronization, structure and boundary spanning.

Decision-making, an important aspect of software development, is the focus of Drury, Conboy and Power's article, "Obstacles to Decision-Making in Agile Software Development Teams". Using a mixed method approach, they investigate decisions involved in iteration planning, execution, review and retrospective, and identify six obstacles to decision-making. They connect the findings to a theory of descriptive decision-making and describe the effects of these obstacles.

Senapathi and Srinivasan focus on the use of agile development methods in the post-adoption stage, in their article, "Understanding Post-Adoptive Agile Usage—An Exploratory Cross-Case Analysis". By adapting theories from systems development and diffusion of innovations, they develop a model that seeks to explain post-adoptive usage of agile practices.

In the article, "Reconciling perspectives: How people manage the process of software development", Adolph and Kruchten develop a grounded theory of social factors in software development. They conceptualize software development as a negotiation process that involves reconciling perspectives, i.e., seeking convergence by sorting out different points of view or perspectives about a software process. Thus, it offers a unique perspective on how agile software development is undertaken in organizations.

A growing interest is evident at agile conferences on identifying ways to combine principles of lean development with software development. In the article "'Leagile' software development: An experience report analysis of the application of lean approaches in agile software development", Wang, Conboy and Cawley distil lessons from 30 experience reports, in six types of lean applications—from practices for continuous process improvement to flow-based development with the Kankan approach.

4. Conclusion

It should be apparent from this introductory article that the research community has lavished attention on the issues related to agile software development ever since the agile manifesto was pronounced in 2001. This is evident from the number of scientific publications, the widespread interest in the topic in various scientific forums, and the number of countries (63) that have been engaged in agile research. The number of special issues devoted

to agile development is also an indication of the keen interest displayed in software engineering and other related fields, notably information systems.

A systematic review of empirical studies published until 2005 (Dybå and Dingsøyr, 2008) called for an increase in both the number and quality of studies. The review also found that most studies focused on eXtreme programming and very few on the Scrum development process, which was gaining significant traction in industry. Further, the review showed the urgent need for more studies involving mature agile development teams, as most studies until then had focused on projects that were just starting to use agile methods.

Has recent progress brought us any closer to a unified framework that brings coherence to the seemingly disparate streams of research being pursued? Our overview of research shows that the number of studies has increased significantly since 2005, and the increased number of journal articles, not just the increased number of conference proceedings, is a sign of increase in quality as well. Going by the attention that they have received, some subfields of agile development appear to be more mature than others. For example, there are meta-studies summarizing experiments on pair programming, with focus on effectiveness (Dybå et al., 2007) and on use in university education (Salleh et al., 2011). Our overview of theories in use in explaining agile software development shows that a range of theories, drawn from many fields, have been applied. This special issue is a further contribution with five articles with strong focus on theory. After the initial spurt of studies on eXtreme programming, the academic community seems to have turned its attention to scrum. Flow-based as well as lean software development has been popular among industry practitioners, but has not yet been extensively researched, as the article by Wang et al. in this issue shows.

Many have called for directions to research on agile development. At *Agile2008*, Dingsøyr et al. (2008) suggested a roadmap for research on agile software development, focusing on providing more empirical research, primarily on experienced agile teams and organizations, connecting better to existing streams of research in more established fields, giving more attention to management-oriented approaches, and finally give more emphasis to core ideas in agile software development in order to increase our understanding.

Ågerfalk et al. (2009) in their introductory article to the special issue on agile/distributed systems development, compiled a top-ten list of future research areas. In the list developed from the inputs of the authors of the special issue articles, the following research areas figured at the top: suitability of agile development to new context such as open *source software* and *software as service*; factors affecting the organizational adaptation of agile methods including tailoring to specific projects; different forms of distributed development and factors facilitating the flexibility, efficiency and effectiveness of such work; ways to extend agile practices beyond software teams into the organizational realm; and identifying boundaries to agile development by applying agility to projects traditionally considered to be non-agile.

At the XP2010 conference, Freudenberg and Sharp (2010) compiled a list of "top ten burning questions" based on feedback from practitioners. Among other issues, they identified agile and large projects, barriers to self-organization, distributed agile, and the role of architecture as high priority topics. At a workshop on new and emerging ideas at *Agile2011*, we posed the question "what should be researched less" and "what should be researched further" to a group mainly consisting of academics. Among other things, they opined that pair programming in educational settings and the reuse of code did not require any further attention. Themes that were deemed to be important included agile across projects and across organizations, the "core" of agile, distributed agile, and the role of architecture and knowledge management in agile development.

We concur that these are exciting research areas that can further our understanding of the effectiveness of agile methods and practices, particularly in different project/organizational contexts. However, our limited analysis of the theoretical perspectives used in prior agile development research suggests that not enough attention is being paid to establishing theoretical underpinnings, when investigating agile development and its various practices. As Jacobson and Spence (2009) point out, sound theoretical roots help us glean the essential concepts, or the "truths" of software development that are methodology-independent. Such theory-driven research enables us to separate true innovations among agile practices from the reinventions and remixes of old approaches, thereby helping us adopt such innovations at a faster rate in the future. Therefore, we urge agile researchers to embrace a more theory-based approach in the future when inquiring into these promising research areas of agile development.

Clearly, the pioneers as well as subsequent researchers of agile development have established a foundation on which the edifice of software development theory and practice can be built. As we stand on the "shoulders of [these] giants" and endeavour to extend the frontiers of software engineering, it is important to remember that the field can mature and progress as a scientific discipline only if efforts are made to provide a robust theoretical scaffold for the conduct of research on agile development. We hope that the articles in this special issue are a step in this direction.

Acknowledgements

We are very grateful to the program committee members of the research at work stage at *Agile2011*, who helped us in inviting authors to revise contributions for this special issue. We also thank the following reviewers: Aybuke Aurum, John McAvoy, Robert Biddle, Philip L. Bond, Nancy Bonner, Judith Brown, Lan Cao, Kieran Conboy, Tore Dybå, Tor E. Fægri, Felix Garcia, Geir Kjetil Hanssen, Børge Haugset, Kishen Iyengar, Philippe Kruchten, Kalle Lyytinen, George Mangalaraj, Kannan Mohan, Maurizio Morisio, Andreas Opdahl, Nilay Oza, Rafael Prikladnicki, Asif Qumer, Bernhard Rumpe, Sherry Ryan, Helen Sharp, Jason Sharp, Vijay Sugumaran, Aakash Taneja, and Vishnu V. Vinekar.

Appendix A. Appendix 1

Rank	Country	Number of publications	%
1	USA	338	21.7
2	CANADA	110	7.1
3	GERMANY	96	6.2
4	FINLAND	94	6.0
5	UK	94	5.3
6	ITALY	56	3.6
7	CHINA	52	3.3
8	IRELAND	51	3.3
9	AUSTRALIA	48	3.1
10	SWEDEN	43	2.8
11	SPAIN	41	2.6
12	NORWAY	33	2.1
13	BRAZIL	29	1.9
14	AUSTRIA	25	1.6
15	ISRAEL	23	1.5
16	NEW ZEALAND	23	1.5
17	POLAND	23	1.5
18	NETHERLANDS	22	1.4
19	CHILE	17	1.1
20	JAPAN	17	1.1
21	DENMARK	16	1.0
22	IRAN	15	1.0
23	INDIA	14	0.9
24	CZECH REPUBLIC	11	0.7

Appendix 1 (*Continued*)

Rank	Country	Number of publications	%
25	SOUTH KOREA	11	0.7
26	ARGENTINA	10	0.6
27	SOUTH AFRICA	10	0.6
28	GREECE	9	0.6
29	PORTUGAL	8	0.5
30	TURKEY	8	0.5
31	FRANCE	7	0.5
32	SWITZERLAND	7	0.5
33	MALAYSIA	6	0.4
34	MEXICO	6	0.4
35	BELGIUM	5	0.3
36	SLOVENIA	5	0.3
37	PAKISTAN	4	0.3
38	ROMANIA	4	0.3
39	TAIWAN	4	0.3
40	BULGARIA	3	0.2
41	CROATIA	3	0.2
42	ESTONIA	3	0.2
43	HUNGARY	3	0.2
44	PHILIPPINES	3	0.2
45	SAUDI ARABIA	3	0.2
46	SINGAPORE	3	0.2
47	COLOMBIA	2	0.1
48	OMAN	2	0.1
49	SYRIA	2	0.1
50	THAILAND	2	0.1
51	U ARAB EMIRATES	2	0.1
52	BOSNIA HERCEG	1	0.1
53	COTE IVOIRE	1	0.1
54	EGYPT	1	0.1
55	ICELAND	1	0.1
56	JORDAN	1	0.1
57	LATVIA	1	0.1
58	LEBANON	1	0.1
59	LITHUANIA	1	0.1
60	MONTENEGRO	1	0.1
61	SERBIA	1	0.1
62	SLOVAKIA	1	0.1
63	ZIMBABWE	1	0.1

References

Abrahamsson, P., Conboy, K., Wang, X., 2009. 'Lots done, more to do': the current state of agile systems development research. European Journal of Information Systems 18, 281–284.

Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J., 2002. Agile software development methods: review and analysis. VTT Technical report, p. 107.

Acuna, S.T., Gomez, M., Juristo, N., 2009. How do personality, team processes and task characteristics relate to job satisfaction and software quality? Information and Software Technology 51, 627–639.

Agarwal, A., Shankar, R., Tiwari, M.K., 2006. Modeling the metrics of lean, agile and leagile supply chain: an ANP-based approach. European Journal of Operational Research 173, 211–225.

Ågerfalk, P., Fitzgerald, B., Slaughter, S., 2009. Introduction to the special issue: flexible and distributed information systems development: state of the art and research challenges. Information Systems Research 20, 317.

Arisholm, E., Gallis, H., Dyba, T., Sjoberg, D.I.K., 2007. Evaluating pair programming with respect to system complexity and programmer expertise. IEEE Transactions on Software Engineering 33, 65–86.

Balijepally, V., Mahapatra, R., Nerur, S., Price, K.H., 2009. Are two heads better than one for software development? The productivity paradox of pair programming. MIS Quarterly 33, 91–118.

Bellini, E., Canfora, G., Garcia, F., Piattini, M., Visaggio, C.A., 2005. Pair designing as practice for enforcing and diffusing design knowledge. Journal of Software Maintenance and Evolution-Research and Practice 17, 401–423.

Boehm, B., 2002. Get ready for agile methods with care. IEEE Computer 35, 64–69.

Boehm, B., Turner, R., 2004. Balancing Agility and Discipline: A Guide to the Perplexed. Addison-Wesley, Boston, MA.

Cao, L., Mohan, K., Xu, P., Ramesh, B., 2009. A framework for adapting agile development methodologies. European Journal of Information Systems 18, 332–343.

Chan, F.K.Y., Thong, J.Y.L., 2009. Acceptance of agile methodologies: a critical review and conceptual framework. Decision Support Systems 46, 803–814.

Choi, K.S., Deek, F.P., Im, I., 2008. Exploring the underlying aspects of pair programming: the impact of personality. Information and Software Technology 50, 1114–1126.

Cockburn, A., 2007. Agile Software Development: The Cooperative Game. Addison-Wesley.

Cohen, D., Lindvall, M., Costa, P., 2004. An introduction to agile methods. In: Zelkowitz, M.V. (Ed.), Advances in Computers, Advances in Software Engineering. Elsevier, Amsterdam.

Conboy, K., 2009. Agility from first principles: reconstructing the concept of agility in information systems development. Information Systems Research 20, 329–354.

Crawford, B., Castro, C., Monfroy, E., 2006. Knowledge management in different software development approaches. In: Yakhno, T., Neuhold, E.J. (Eds.), Advances in Information Systems, Proceedings. , pp. 304–313.

Dingsøyr, T., Dybå, T., Abrahamsson, P.,2008. A Preliminary Roadmap for Empirical Research on Agile Software Development. In: in Proc. of Agile2008. IEEE Press, pp. 83–94.

Dingsøyr, T., Dybå, T., Moe, N.B., 2010. Agile Software Development: Current Research and Future Directions. Springer, Berlin/Heidelberg.

Dingsøyr, T., Hanssen, G.K., 2002. Extending agile methods: postmortem reviews as extended feedback. In: Henninger, S., Maurer, F. (Eds.), Advances in Learning Software Organizations. , pp. 4–12.

Doran, H.D., 2004. Agile knowledge management in practice. In: Melnik, G., Holz, H. (Eds.), Advances in Learning Software Organizations, Proceedings. , pp. 137–143.

Dybå, T., 2011. Special section on best papers from XP2010. Information and Software Technology 53, 507–508.

Dybå, T., Arisholm, E., Sjøberg, D.I.K., Hannay, J.E., Shull, F., 2007. Are two heads better than one? On the effectiveness of pair programming. IEEE Software 24, 12–15.

Dybå, T., Dingsøyr, T., 2008. Empirical studies of agile software development: a systematic review. Information and Software Technology 50, 833–859.

Erdogmus, H., Morisio, M., Torchiano, M., 2005. On the effectiveness of the test-first approach to programming. IEEE Transactions on Software Engineering 31, 226–237.

Erickson, J., Lyytinen, K., Siau, K., 2005. Agile modeling, agile software development, and extreme programming. Journal of Database Management 16, 88–100.

Falessi, D., Cantone, G., Sarcia, S.A., Calavaro, G., Subiaco, P., D'Amore, C., 2010. Peaceful coexistence: agile developer perspectives on software architecture. IEEE Software 27, 23–25.

Fang, M., Ying, J., Wu, M.H., 2004. Effective elements of integrated software development process supported platform. In: Shen, W., Lin, Z., Barthes, J.P.A., Li, T. (Eds.), Computer Supported Cooperative Work in Design I. , pp. 368–377.

Freudenberg, S., Sharp, H., 2010. The top 10 burning research questions from practitioners. IEEE Software 27, 8–9.

Hannay, J.E., Arisholm, E., Engvik, H., Sjoberg, D.I.K., 2010. Effects of personality on pair programming. IEEE Transactions on Software Engineering 36, 61–80.

Hazzan, O., Dubinsky, Y., 2005. Social perspective of software development methods: the case of the prisoner dilemma and extreme programming. In: Baumeister, H., Marchesi, M., Holcombe, M. (Eds.), Extreme Programming and Agile Processes in Software Engineering, Proceedings. , pp. 74–81.

Henderson-Sellers, B., Serour, M.K., 2005. Creating a dual-agility method: the value of method engineering. Journal of Database Management 16, 1–23.

Highsmith, J., Cockburn, A., 2001. Agile software development. 1. The business of innovation. IEEE Computer 34, 120–127.

Holz, H., Maurer, F., 2002. Knowledge management support for distributed agile software processes. In: Henninger, S., Maurer, F. (Eds.), Advances in Learning Software Organizations. , pp. 60–80.

Jacobson, I., Spence, I., 2009. Why we need a theory for software engineering. Dr. Dobb's Journal.

Janzen, D., Saiedian, H., 2005. Test-driven development concepts, taxonomy, and future direction. Computer 38, 43–50.

Johannessen, L.K., Ellingsen, G., 2009. Integration and generification-agile software development in the healthcare market. Computer Supported Cooperative Work-the Journal of Collaborative Computing 18, 607–634.

Layman, L., Williams, L., Slaten, K., Berenson, S., Vouk, M., 2008. Addressing diverse needs through a balance of agile and plan-driven software development methodologies in the core software engineering course. International Journal of Engineering Education 24, 659–670.

Lee, G., Xia, W., 2010. Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. MIS Quarterly 34, 87–114.

Levardy, V., Browning, T.R., 2009. An adaptive process model to support product development project management. IEEE Transactions on Engineering Management 56, 600–620.

Lyytinen, K., Rose, G.M., 2006. Information system development agility as organizational learning. European Journal of Information Systems 15, 183–199.

Mafakheri, F., Nasiri, F., Mousavi, M., 2008. Project agility assessment: an integrated decision analysis approach. Production Planning & Control 19, 567–576.

Mangalaraj, G., Mahapatra, R., Nerur, S., 2009. Acceptance of software process innovations—the case of extreme programming. European Journal of Information Systems 18, 344–354.

McAvoy, J., Butler, T., 2007. The impact of the Abilene Paradox on double-loop learning in an agile team. Information and Software Technology 49, 552–563.

McCain, K.W., 1990. Mapping authors in intellectual space: a technical overview. Journal of the American Society for Information Science 41, 433–443.

Meso, P., Jain, R., 2006. Agile software development: adaptive systems principles and best practices. Information Systems Management 23, 19–30.

Moe, N.B., Dingsøyr, T., Dybå, T., 2009. Overcoming barriers to self-management in software teams. IEEE Software 26, 20–26.

Moe, N.B., Dingsøyr, T., Dybå, T., 2010. A teamwork model for understanding an agile team: a case study of a Scrum project. Information and Software Technology 52, 480–491.

Nawrocki, J., Wojciechowski, A., 2001. Experimental evaluation of pair programming. In: 12th European Software Control and Metrics Conference, ESCOM, London, UK, pp. 269–276.
Nerur, S., Balijepally, V., 2007. Theoretical reflections on agile development methodologies. Communications of the ACM 50, 79–83.
Nerur, S., Mahapatra, R., Mangalaraj, G., 2005. Challenges of migrating to agile methodologies. Communications of the ACM 48, 73–78.
Nerur, S.P., Rasheed, A.A., Natarajan, V., 2008. The intellectual structure of the strategic management field: an author co-citation analysis. Strategic Management Journal 29, 319–336.
Northover, M., Boake, A., Kourie, D.G., 2006. Karl Popper's critical rationalism in agile software development. In: Scharfe, H., Hitzler, P., Ohrstrom, P. (Eds.), Conceptual Structures: Inspiration and Application. , pp. 360–373.
Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., Still, J., 2008. The impact of agile practices on communication in software development. Empirical Software Engineering 13, 303–337.
Ramesh, B., Cao, L.A.N., Mohan, K., Peng, X.U., 2006. Can distributed software development be agile? Communications of the ACM 49, 41–46.
Ramos-Rodríguez, A.-R., Ruíz-Navarro, J., 2004. Changes in the intellectual structure of strategic management research: a bibliometric study of the Strategic Management Journal, 1980–2000. Strategic Management Journal 25, 981–1004.
Salazar-Torres, G., Colombo, E., Da Silva, F.S.C., Noriega, C.A., Bandini, S., 2008. Design issues for knowledge artifacts. Knowledge-Based Systems 21, 856–867.
Salleh, N., Mendes, E., Grundy, J., 2011. Empirical studies of pair programming for CS/SE teaching in higher education: a systematic literature review. IEEE Transactions on Software Engineering 37, 509–525.
Sena, J.A., Shan, A.B., 2002. Integrating knowledge management, learning mechanisms, and company performance. In: Karagiannis, D., Reimer, U. (Eds.), Practical Aspects of Knowledge Management. , pp. 620–631.
Sfetsos, P., Stamelos, I., Angelis, L., Deligiannis, I., 2006. Investigating the impact of personality types on communication and collaboration-viability in pair programming—an empirical study. In: Abrahamsson, P., Marchesi, M., Succi, G. (Eds.), Extreme Programming and Agile Processes in Software Engineering, Proceedings. , pp. 43–52.
Sfetsos, P., Stamelos, I., Angelis, L., Deligiannis, I., 2009. An experimental investigation of personality types impact on pair effectiveness in pair programming. Empirical Software Engineering 14, 187–226.
Sharp, H., Robinson, H., 2006. A distributed cognition account of mature XP teams. In: Abrahamsson, P., Marchesi, M., Succi, G. (Eds.), Extreme Programming and Agile Processes in Software Engineering. Springer, Berlin/Heidelberg, pp. 1–10.
Sharp, H., Robinson, H., 2008. Collaboration and co-ordination in mature eXtreme programming teams. International Journal of Human-Computer Studies 66, 506–518.
Sharp, H., Robinson, H., Petre, M., 2009. The role of physical artefacts in agile software development: two complementary perspectives. Interacting with Computers 21, 108–116.
Siau, K., 2005. A retrospective review of JDM from 2003 to 2005 and a discussion on publication emphasis of JDM for the next two to three years. Journal of Database Management 16, 1.
Socha, D., Walter, S., 2006. Is designing software different from designing other things? International Journal of Engineering Education 22, 540–550.
Trinidad, P., Benavides, D., Duran, A., Ruiz-Cortes, A., Toro, M., 2008. Automated error analysis for the agilization of feature modeling. Journal of Systems and Software 81, 883–896.
White, H.D., 1990. Author co-citation analysis: overview and defense. In: Borgman, C.L. (Ed.), Scholarly Communication and Bibliometrics. Sage Publications, Newbury Park, pp. 84–106.
White, H.D., Griffith, B.C., 1981. Author cocitation: a literature measure of intellectual structure. Journal of the American Society for Information Science 32, 163–171.
Williams, L., Cockburn, A., 2003. Agile software development: it's about feedback and change. Computer 36, 39–43.
Williams, L., Kessler, R.R., Cunningham, W., Jeffries, R., 2000. Strengthening the case for pair programming. IEEE Software 17, 19–25.
Zimmer, J.A., 2003. Graph theoretical indicators and refactoring. In: Maurer, F., Wells, D. (Eds.), Extreme Programming and Agile Methods. XP/Agile Universe 2003, pp. 62–72.

Torgeir Dingsøyr works with software process improvement and knowledge management projects as a senior scientist at SINTEF Information and Communication Technology. In particular, he has focused on agile software development through a number of case studies, co-authoring of a systematic review of empirical studies, co-editing of the book *Agile Software Development: Current Research and Future Directions*, and was co-organizing chair of the *11th International Conference on Agile Software Development* (XP2010) as well as co-producer of the research at work stage at *Agile2011*. He wrote his doctoral thesis on *Knowledge Management in Medium-Sized Software Consulting Companies* at the Department of Computer and Information Science, Norwegian University of Science and Technology, where he is now adjunct associate professor.

Sridhar Nerur is an associate professor of Information Systems at the University of Texas at Arlington. He holds an engineering degree in electronics from Bangalore University, a PGDM (MBA) from the Indian Institute of Management, Bangalore, India, and a PhD in business administration from the University of Texas at Arlington. His publications include articles in leading journals such as *MIS Quarterly*, *the Strategic Management Journal*, *Journal of International Business Studies*, *Communications of the ACM*, *Communications of the AIS*, *the DATA BASE for Advances in Information Systems*, *European Journal of Information Systems*, and *Information Systems Management*. He served as an associate editor of the *European Journal of Information Systems*. His research and teaching interests are in the areas of software design, adoption of software development methodologies, cognitive aspects of programming, dynamic IT capabilities, and agile software development.

Venu Gopal Balijepally is an associate professor of MIS in the College of Business at Prairie View A&M University, Texas. He received his PhD in information systems from the University of Texas at Arlington and post-graduate diploma in management (MBA), from the Management Development Institute, Gurgaon, India. He also holds a masters degree from Indian Institute of Technology, Mumbai and a bachelor's degree from Osmania University, India, both in civil engineering. His research interests include software development, social capital of IS teams, knowledge management and IT management. His research publications appear in *MIS Quarterly*, *Journal of International Business Studies*, *Journal of the AIS*, *Communications of the ACM*, *Communications of the AIS*, and various conference proceedings such as the Americas Conference on Information Systems, the Hawaii International Conference on System Sciences, and the Decision Sciences Institute.

Nils Brede Moe works with software process improvement, agile software development and global software development as a senior scientist at SINTEF Information and Communication Technology. His research interests are related to organizational, socio-technical, and global/distributed aspects. His main publications in the field of agile software development include several longitudinal studies on self-management and teamwork, and co-editing of the books *Agile Software Development: Current Research and Future Directions* and *Agility Across Time and Space: Implementing Agile Methods in Global Software Projects*. Moe was also a co-organizing chair of the *11th International Conference on Agile Software Development* (XP2010) and co-producer of the research at work stage at *Agile2011* and *Agile India 2012*. He wrote his thesis for the degree of doctor philosophiae on *From Improving Processes to Improving Practice—Software Process Improvement in Transition from Plan-driven to Change-driven Development*. He is currently a visiting researcher at the University of New South Wales, Sydney.

Torgeir Dingsøyr[a,b,*]
Sridhar Nerur[c]
VenuGopal Balijepally[d]
Nils Brede Moe[a]
[a] *SINTEF, NO-7465 Trondheim, Norway*
[b] *Norwegian University of Science and Technology, Department of Computer and Information Science, Sem Sælandsvei 7-9, NO-7491 Trondheim, Norway*
[c] *Department of Information Systems and Operations Management, University of Texas at Arlington, Arlington, TX 76019, USA*
[d] *Department of Accounting, Finance & MIS, Prairie View A&M University, Prairie View, TX 77446-0519, USA*

[*] Corresponding author at: SINTEF, NO-7465 Trondheim, Norway. Tel.: +47 93008714.
E-mail addresses: torgeir.dingsoyr@sintef.no (T. Dingsøyr), snerur@uta.edu (S. Nerur), vebalijepally@pvamu.edu (V. Balijepally), nils.b.moe@sintef.no (N.B. Moe)

12 February 2012

13 February 2012
Available online 7 March 2012