Association for Information Systems
AIS Electronic Library (AISeL)

Issue Nr 7 (2016) Scandinavian (IRIS)

2016

How has Agile Methods Inspired an Industrywide Project Management Initiative?

Lise T. Heeager
Aarhus University, Denmark, lith@mgmt.au.dk

Per Svejvig
Aarhus University, Denmark, psve@mgmt.au.dk

Bjarne Rerup Schlichter
Aarhus University, Aarhus, Denmark, brs@mgmt.au.dk

Follow this and additional works at: http://aisel.aisnet.org/iris2016

Recommended Citation
Heeager, Lise T.; Svejvig, Per; and Schlichter, Bjarne Rerup, "How has Agile Methods Inspired an Industrywide Project Management Initiative?" (2016). *Issue Nr 7 (2016)*. 7.
http://aisel.aisnet.org/iris2016/7

This material is brought to you by the Scandinavian (IRIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in Issue Nr 7 (2016) by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

How has Agile Methods Inspired an Industrywide Project Management Initiative?

Lise Tordrup Heeager[1], Per Svejvig[1]and Bjarne Rerup Schlichter[1],

[1] Aarhus University, Department of Management, Bartholin's Allè 10,
8000 Aarhus C, Denmark
{lith, psve, brs}@mgmt.au.com

Abstract. Increased complexity in projects has forced new project management initiatives. In software development several agile methods have emerged and methods such as Scrum are today highly implemented in practice. General project management practice has been inspired by agile software development. But in order to fully understand and to provide suggestions for future practice on how agility can be incorporated in general project management, this paper addresses how agile methods have inspired general project management practices. To answer the research question, the paper provides an analysis which compares ten characteristics of agile software development (identified in theory) and the general project management method developed by the Danish Project Half Double (PHD) initiative. The method consists of 10 leading stars (principles) and the impact, flow and leadership (IFL) method for rethinking project management. The analysis showed how general project management to a large degree has been conquered by agile methods.

Keywords: Agility**,** Agile Software Development, General Project Management**,** the Danish Project Half Double (PHD).

1 Introduction

Newer ideas on management of projects introduce a plenteous of approaches, i.e. by understanding the project as a temporary organization, where learning, diversity, temporality, complexity, uncertainty and sociability are in play [1, 2]. The increased complexity of projects has led to the development of agile methods and to the acceptance of the need for a contemporary organization of projects that not only mechanically executes a process towards a narrow, product-oriented goal, but accepts projects as something business-like and value-creating [3].

Agile methods, such as Scrum, are highly used in the development of software, based on a belief that a project in broad understanding continuously needs to be adjusted based on the learning acquired during the process. The use of Agile methods in software development settings seems to have a greater impact on success factors than just straight efficiency [4].

Traditionally a project has been seen as a tool applied to a single assignment, focusing on meeting time and quality under the available resources. This single-track approach applies a logical and chronological way through a set of (more or less) well-defined tasks. This approach has been challenged in the general understanding on how to modernize project management, especially in the stream of research related to Rethinking Project Management [2].

But how can agile methods be applied in practice on projects outside of the software development domain as a more generic concept? Adopting agile methods in general project management settings requires a change from command and control management to leadership and collaboration [5], [6], which requires a reorientation not only for the project team but also from management [7].

It is claimed that reorientation and construction of these above mentioned management models requires a deep understanding of the "agility" construct in management practices [8], which has been the inspiration for the present study and leading to the following research question: *How has Agile Software Development methods inspired general project management practices?*

We pursue to answer the research question by drawing on the Danish Project Half Double (PHD) initiative. This is an industry-driven initiative by a consultancy firm involving several private and public organizations including three universities [9]. PHD has produced 10 leading stars (principles) for rethinking project management inspired by similar work about a new mindset for management [10], but also highly inspired by agile thinking [11]. PHD has a profound desire to change the practices in projects and project management much in line with the agile manifesto [12]. The study is conducted as collaborative research [13, 14] where practitioners and researchers share ideas and are involved in activities to co-produce knowledge about the PHD initiative. Research about agile methods is used as a theoretical lens to understand and explain aspects of the PHD initiative.

The remainder of this paper is organized as follows. The next section presents a brief summary of the characteristic of Agile Software Development. The research setting and approach are then described as well as data collection and data analysis. The paper continues with a section presenting the empirical data, followed by the comparative analysis presenting similarities and differences between Agile Software Development and Project Management Practices from the Project Half Double case providing knowledge on how Agile Software Development has inspired and enriched general Project management Practices. The paper ends with the concluding remarks and suggestions for further research.

2 Agile Software Development

Agile software development methods promise a way to deliver software without excessive cost [15]. They were derived from the weaknesses and failures of the traditional, plan-driven methods [16]. In 2001, the agile manifesto was created by a group of practitioners; it constitutes four values and a set of practices for software development. The four values are: 1) individuals and interactions over processes and tools 2) working software over comprehensive documentation 3) customer

collaboration over contract negotiation 4) responding to change over following a plan [12]. Through an iterative, test-first software development strategy with frequent customer feedback, the agile methods seek high software quality. The short iterations, multi-disciplinary teams, knowledge sharing and continuous integration allow better control over the project and increase visibility [17]. To enhance knowledge sharing, they advocate an informative workspace that includes information radiators [18]. Agile methods rely on the competencies of the software developers [19]. Refactoring (redesigning and rewriting software) is also advocated by the agile methods, as it serves to correct poorly written and redundant code [20]. This paper adopts the definition of agility provided by Conboy [21]; it takes its starting point in the concepts of flexibility and leanness:

> *"the continual readiness of an ISD method 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." (Conboy 2009, p. 340).*

Examples of agile methods are Scrum [22] and eXtreme Programming (XP) [23]. Today, these are the two most well-known and popular agile methods. While XP offers a range of practices and principles to apply [23] in order to create flexibility and adaptability, Scrum provides guidance for the efficient management of projects [24]. As this paper is concerned with project management Scrum will be described briefly [22]. Scrum includes three central roles: the Scrum master, the product owner and the team. The Scrum master is responsible for managing the software development practice, making sure that the Scrum values, practices and rules are enforced. The product owner is officially responsible for the project, and ideally this role is filled by the customer. The Scrum team is the developers and testers and other people working on the software. It furthermore consists of the practices: sprints, daily Scrum meetings, sprint planning meetings, sprint reviews and the backlogs. A sprint is a fixed period of time for which the team works; Scrum recommends 30 day sprints in which there is no interference. Every day a Scrum meeting is held; it is a status meeting of a maximum of 15 minutes. At these meetings the managers can get a sense of what and how the team is doing. At the sprint planning meeting the functionality of the next sprint is determined. The sprint review is a meeting in which the Scrum team presents what has been built during the sprint. Scrum operates with three backlogs, the product backlog (all product requirements), the release backlog (the tasks chosen for the next release) and the sprint backlog (containing the tasks for the current sprint).

2.1 Ten characteristics of Agile Software Development

Based on a review of the classical agile literature, the practices from the above mentioned agile methods have been synthesized into ten characteristics of agile software development.

1) Short iterations of learning: Dealing with change is imperative in software development [25]. To ensure adaptability, agile methods advocate an iterative development strategy [12], that include all phases of the development process, analysis, design, implementation, test and evaluation [19]. The goal of the short iterations is to give the developers a time to work without interruptions and to include flexibility and support the learning that happens during a project [22]. Over time the complexity of the product tend to grow, in XP a principle is to refactor - to chip away complexity while still delivering at the end of the iteration [23].

2) Frequent continuous customer involvement: The customer plays a central role in agile development and the customer is involved to a great extent [26]. In agile projects the customer plays an active role throughout the whole project-life-cycle and feedback is provided continuously both to developers and back to customers [27]. The customer involvement in agile methods is therefore much greater. XP recommends an on-site customer [23], which in practice has not been widely adopted [28]. In Scrum the customer is involved through the product owner role [22].

3) Teams working closely together: Agile team are small (approximately 7 people) in order to be able to work together closely. XP furthermore advocates that the team sit together on a daily basis and use team activities such as pair programming for the development [23]. Placing people together help them respond to change [27].

4) Person-to-person informal coordination and openness: The agile methods rely on person-to-person communication and knowledge sharing and advocates openness; in the terms used by Hansen, Nohria [29], the agile methods primarily rely on a personalization strategy. The frequent meetings (stand-up meetings, planning meetings and retrospectives) proposed by the agile methods serve as ways to informally communicate and share knowledge, enhanced by the self-managing developer teams [30]. Furthermore, information on the project and its status must be provided in the open space, this could for example include task boards and burn down charts placed where passersby can see them. [18].

5) Self-managing teams: The overall management style in agile software development is leadership and collaboration [5]. An essential part of this is implementing self-managing teams that can organize in various configurations [27]. To implement this practice, it is not enough to put people together and label them self-managing, people need training in how to coordinate and work effectively as an agile team [7]. Education is vital as both the developers and the management need to become familiar with the agile principles and practices before attempting to adopt its practices. Such training is used to eliminate misconceptions and tackle the fear of change. In Scrum, the scrum master has a leadership role, for the self-managing teams. A focus on the team and how the scrum master can help the team achieve the sprint goal is central. The scrum master is, for example, to make sure the team gets to work focused on the project during the sprint [22].

6) Motivation through collective ownership: in agile software development all team members are responsible for reaching the goal of the iteration. In principle every team member can work on any task assigned to the iteration. Working in self-organizing teams with common responsibility for tasks [22] and code [23] enhances team motivation.

7) Early focus on the product: One of the agile principles is: “working software over comprehensive documentation”. As the manifest suggest; all though agile

methods do not focus mostly on documentation, they do not cast all documentation aside [31]. The focus is however on having at least a part of the system/product ready early in the process [32].

8) From user-stories to the product: for specifying the requirements, agile methods rely primarily on user stories written by the customer in a plain business-like language [33]. The stories must be displayed in the work area [18].

9) Test-first strategy: the test-first strategy has become increasingly popular within the agile community [6]. In XP for example, testing is a core practice; it is not only suggested that test cases are written before the software, it is also more importantly suggested that all parts of an increment is tested and that completion is determined by the increment passing all tests [23].

10) People-aspects: The agile methods focus on the social aspects of software development [34] and people-issues are at the heart of the agile movement [35]. This is for example practiced through self-managing teams and frequent customer involvement. This focus on people and less focus on processes poses new requirements for the qualifications of the project members. Attention therefore needs to be given to people-related factors such as staffing and communications [36]. Amicability, talent, skill and communication are critical people factors for agile methods, as proposed by [27].

3 Research Setting and Research Method

The Project Half Double initiative was started in 2013 as an informal network of very committed people at different levels from Danish industry who discussed how to develop project management in the light of the apparent high failure rate of projects (e.g. CHAOS Reports [37, 38] and other studies claiming high failure rates), and with the ambition to manage projects in a radically different way. The initiative was centered on “Implement Consulting Group” (hereafter Implement), a Scandinavian-based management consultancy company, with more than 450 consultants on board, with global reach that helps organizations change. The initiative matured and began gradually to formalize during spring 2014 and the work manifested into the 10 leading stars, which have been developed and discussed at four workshops from February 2014 to January 2015 with a broad representation from areas such as manufacturing, finance, insurance, IT, public administration, management consultancies, universities and the Confederation of Danish Industry. The 10 leading stars have later developed into a more fine grained method called Impact, Flow and Leadership (IFL) method. The formal project kick-off took place in June 2015 where seven pilot projects from seven industry organizations was started in order to test the PHD method, (consisting of both the 10 leading stars and the ILF method) and this will last until summer 2016.

This paper is focused on comparing the PHD method with the agile research stream, and not the larger scope of PHD. The specific research design for this paper is qualitative comparative research [39]. The primary data collection methods included participation in workshops and meetings from February 2014 to October 2015 with pilot organizations, the Danish Industry Foundation and Implement. The workshops and meetings are documented by written material, videos and field notes taken by the

authors. There are two artifacts that are particularly relevant for the analysis in this paper: (1) the 10 leading stars (finalized June 2015), and (2) the impact, flow and leadership method (March 2016). Informal talks with members of the PHD network have also broadened the understanding of PHD thinking and how practitioners relate this to their lifeworld [40, 41]. This paper has adopted the research design from Svejvig and Grex [9].

4 Project Half Double Method with the 10 Leading Stars and the Impact, Flow and Leadership Method

The 10 leading stars thus grew gradually out of the pre-project activities that took place from spring 2013 to spring 2015 and could be described as the PHD intellectual foundation, and they are elaborated below:

Leading star no. 1 – Focus on customer value: Focus on project benefits not on the execution model. A project is a pitch into the future. You no longer want "business as usual," you want to create something new that will impact the surrounding environment. This is why the project leader, the project team, the steering committee and everybody with a hand in the project must maintain focus on the value that the project is actually creating.

Leading star no. 2 – Put people before execution models: Human behavior is not a mathematical statement that can be solved by means of intricate models. The prevailing execution models often see projects as linear systems aimed at respecting the rules and phases as strictly as possible to reach the desired results. But a project is a social entity that involves people with different professional expertise, experience and personalities, and who even may be located in different parts of the world.

Leading star no. 3 – Colocation: The right people do not only work on the same project – they collaborate. Project work should be about collaboration in the fundamental sense of the word. A supply flow where one specialist solves his part of the task and sends on the project to the next one is not collaboration. On the contrary, it is a throwback to the days of assembly lines.

Leading star no. 4 – Leadership is hard-core trust: Hard-core trust is superior to toughness and trust separately. Future leaders of project organizations are tough but trust inspiring. They focus on the objective and less on telling people what to do to reach the objective. In fact, the best leaders have an exceptional focus on creating fabulous and unique solutions that are delivered on time. They manage to do that because, on the one hand, they are tough when making demands on their project workers while at the same time fully trusting them to live up to the demands.

Leading star no. 5 – Lead inwards: If you want to create results as a leader, use your energy in the project. Successful projects require leadership. Therefore a project organization has no use for leaders who focus too much on leading outwards or upwards. A project organization must have leaders who have a burning passion for leading projects.

Leading star no. 6 – From steering committee to chaos committee: The term "steering committee" indicates that a group above the project can steer it. In the real world this is not so, because it is impossible to steer a project top-down. If a steering

committee wants to create value for a project, it must be used for other things than making go/no-go decisions. The project leader must include the steering committee in discussions on the true challenges of the project. One way of ensuring this is by relabeling them from steering committees to chaos committees and viewing them as a forum where the project can get mentoring, wild ideas and external perspective.

Leading star no. 7 – Quick insight: Effective project execution amounts to a steep learning curve. In all projects we learn something new right up to the time where the project is concluded and the result launched. The awareness we reach along the way helps guide the project – often in a different direction than anticipated. This may imply delays and increase costs but it is also a necessary aspect of development work.

Leading star no. 8 – Short and fat projects: Allocate fewer people with more time. Many organizations like to run more projects than their resources actually permit. Often this implies that each project is allocated fewer resources but in turn a longer time. As a consequence, projects are long (throughput time) and thin (resource allocation), which will affect the quality of the project.

Leading star no. 9 – Work with visuals: Make it easy and intuitive to share insight. Visual communication is an important tool in modern project work where there is a need to share knowledge in a quick and intuitive way. Instead of spreadsheets and diagram communication, a large visible plan can be an important tool when you have to reach an agreement on goals and work processes.

Leading star no. 10 – Kill complexity: Focus on simplicity in solutions, not complexity in organizations. The simplest solution is often the best one but many solutions end up being rather complex. Most people have a need to flaunt their professionalism. Often the simple solution is neglected because people worry about not seeming clever enough or professionally competent. Simple solutions require guts.

The 10 leading stars served as a good starting point for PHD, but they were also at too high a level and therefore not operational enough to be used directly in projects, so a practice-oriented method was needed. The consultants thus developed the "impact, leadership and flow method" (IFL method), which is described below:

Impact is the essential keyword in PHD, where it is understood as a synonym for value [42]. Creating impact is all about time to impact, focus on value creation and stakeholder satisfaction. There is an immense focus on reducing time to impact, and how impact should drive the project. Energy amongst key stakeholders should be established in order to support project impact, and stakeholder satisfaction has to be the ultimate evaluation criterion and measured frequently (pulse check). One statement from the IFL method is "*The new project triangle is circular with impact in the center*," which metaphorically indicates the move from the triple constraint (iron triangle with scope, time and cost) [43] to something different. The IFL method suggests several tools for working with impact in projects such as "impact case with success criteria" and "monthly pulse check" supported by impact reinforcement activities.

F**low:** The focus is on flow efficiency instead of resource efficiency. However, the flow focus will imply two opposing forces, which have to be balanced: (1) Increase freedom where the project team is free to solve the task how they see fit, they can build their own infrastructure, and they decide how to act on feedback; (2) Increase

restriction with fixed time boxes of one month (sprint thinking), no scope changes during sprints and structured feedback, daily and weekly (scrum and sprint meetings) [see 11]. The flow part is supported by five key events (sprint planning, visual status etc.), five clear roles such as project owner, five non-negotiable rules where, for example, "*team members should be allocated more than 60% on a given project during spring cycles*," and finally five values (learn from reality, keep it simple etc.). Visual planning is a foundation for the IFL method as seen in agile thinking with sprint boards (for example a flip chart with post is categorized into to do, doing and done), and front-loading is used as a way to accelerate the knowledge where front-loading is defined as a "*problem-solving…strategy that seeks to improve development performance by shifting the identification and solving of [design] problems to earlier phases of a product development process* [44].

Leadership is the least developed discipline within the IFL method, but it deals with executive management where the "*chief executive owns impact and the project owner is active and close to the project*," labeled active project ownership. This is further elaborated into the idea that an active project owner has an informal and trusted relationship with the project manager similar to the dynamic duo (metaphorically described as the partnership between the superheroes Batman and Robin). The project owner should allocate sufficient time to help the project and project manager whereby the rule is that an active project owner participates in a maximum of three steering committees at the same time. The next level of management is the project manager, who should be a firm project manager with authority and business focus. The project manager has to be inspirational and dare to persuade people to believe in the same dream about the project – a sense giver. The project manager should have a business focus articulated with the sentence "show me the money."

The paper continues with a comparison of agile methods with the 10 leading stars and the IFL method.

5 Comparing Agile Methods with Project Half Double Method

Before presenting the comparison it is important to emphasize that agile methods for software development are linked to the specific domain of software development while the 10 leading stars and IFL method were developed to general project management and not linked to a particular project domain. Some translation has therefore been necessary between the specific software domain and more general project domains. Table 1 below shows the comparison:

Table 1. Comparing the ten characteristics of agile software development with Project Half Double Method

No.	Ten characteristics of agile software development	Influence on the Project Half Double Method	Ful-filled
1	Short iterations of learning	The IFL method includes fixed project pace with compressed sprints. Leading star no. 7 quick insight focus on enabling a steep learning curve in projects although this is not short iterations of learning it shares the idea of focus on learning	Yes
2	Frequent continuous customer involvement	Leading star no. 1 states focus on customer value, and this requires typically continuous customer involvement to have this value focus. The IFL method also states that project owner is highly involved in the project	Yes
3	Teams working closely together	Leading star no. 3 is about colocation, which is about teams working closely together. The IFL method furthermore states that all core team member should be allocated more than 60% to the project	Yes
4	Person-to-person informal coordination and openness	Colocation as mentioned above stimulates informal coordination and openness. Leading star no. 9 about working with visuals do also fertilize for person-to-person discussion e.g. by discussing planning or other issues at the visual information radiator in the room	Yes
5	Self-managing teams	*Not directly addressed in PHD method*	*No*
6	Motivation through collective ownership	*Not directly addressed in PHD method*	*No*
7	Early focus on the product	The IFL method and 10 leading has a profound focus on impact and value while product is a mean to achieve the goal. However the five key events and the five values in the IFL method does indirectly support focus on early product creation	Partly
8	From user-stories to the product	*Not directly addressed in PHD method*	*No*
9	Test-driven development	The IFL method with the five key events with sprint thinking does indirectly imply focus on testing and discussing the product on a regular basis	Partly
10	People-aspects: focus on individuals over processes	Leading star no. 2 about put people before execution models maps nicely to this characteristics	Yes

The comparison between agile methods and PHD method is rather high level and an analysis at a more detailed level might give more fine-grained similarities and differences, but this high level comparison does in itself emphasize that agile methods from software development certainly has diffused into general project management practices at least in this specific example of PHD, but other studies support the same tendency [4, 8].

Most of the ten characteristics are mirrored in the PHD method are either fully or partly fulfilled. The remaining three characteristics are relevant to consider for PHD although e.g. self-managing teams implies a fairly different leadership style than proposed by the PHD method. Agile methods for software development might on the other hand also be inspired by the PHD method as there are several leading stars not mapping directly to the agile methods such as leadership is hard-core trust (no. 4) and short and fat projects (no. 8).

5 Concluding Remarks

This paper answers the research question: *How has Agile Software Development methods inspired general project management practices?* by identifying how the ten characteristics of agile software development has influenced the Project Half Double Method. It is seen that most of the characteristics are mirrored in the PHD method. The PHD method studied in this paper is a specific example of a new project management method which is making its way to practice. However, since the method is developed by a large group of practitioners summarizing their best practices and since the method now has been diffused into several organizations, we regard the method as a valid representative for how project management is practiced today. For organizations that are not following the PHD method or a similar method and wanting to increase agility, we can recommend to consider the PHD method.
This conceptual mapping opens a set of interesting avenues for further research. Among these are looking deeper into specific aspects of the IFL method and how this can be mapped into an agile approach or looking deeper into more specific types of agile development, such as XP or Scrum.

We acknowledge the limitations of the present research and understands that the conceptualization must be challenged further by applying it on more cases and by analyzing both the small discrepancies and how the actors understands and applies these agile principles in their daily project practices.

Acknowledgement

The authors would like to thank the Danish Industry Foundation for funding this work, and acknowledge contributions from Danish organizations involved in Project Half Double and Implement Consulting Group. The authors declare that they have no conflict of interest regarding the funding agency and other parties involved in Project Half Double.

References

1. Winter, M. and T. Szczepanek, *Images of projects*. 2009: Gower Publishing, Ltd.

2. Svejvig, P. and P. Andersen, *Rethinking project management: A structured literature review with a critical look at the brave new world*. International Journal of Project Management, 2015. **33**(2): p. 278-290.
3. Davenport, T.H., *Process innovation: reengineering work through information technology*. 2013: Harvard Business Press.
4. Serrador, P. and J.K. Pinto, *Does Agile work? — A quantitative analysis of agile project success.* International Journal of Project Management, 2015. **33**(5): p. 1040-1051.
5. Misra, S., V. Kumar, and U. Kumar, *Identifying some critical changes required in adopting agile practices in traditional software development projects.* International Journal of Quality & Reliability Management, 2010. **27**(4): p. 451-474.
6. Nerur, S., R. Mahapatra, and G. Mangalaraj, *Challenges of migrating to agile methodologies.* Communications of the ACM, 2005. **48**(5): p. 73-78.
7. Moe, N.B., T. Dingsøyr, and T. Dybå, *A teamwork model for understanding an agile team: A case study of a Scrum project.* Information and Software Technology, 2010. **52**(5): p. 480-491.
8. Conforto, E.C., et al., *Can Agile Project Management Be Adopted by Industries Other than Software Development?* Project Management Journal, 2014. **45**(3): p. 21-34.
9. Svejvig, P. and S. Grex, *The Danish Agenda for Rethinking Project Management.* International Journal of Managing Projects in Business, 2016. **9**(4): p. 822-844.
10. Hamel, G., *Moon Shots for Management.* Harvard Business Review, 2009. **87**(2): p. 91-98.
11. Schwaber, K., *Agile project management with Scrum*. 2004: Microsoft Press Redmond, WA, USA.
12. Beck, K., et al. *Manifesto for Agile Software Development*. 2001 [cited 2015.
13. Mathiassen, L., *Collaborative practice research.* Information Technology & People, 2002. **15**(4): p. 321-345.
14. Van de Ven, A., *Engaged scholarship: A guide for organizational and social research*. 2007, Oxford: Oxford University Press.
15. Boehm, B. and R. Turner. *Balancing agility and discipline: Evaluating and integrating agile and plan-driven methods*. in *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on*. 2004. IEEE.
16. Taylor, P., et al., *Applying an agility/discipline assessment for a small software organisation*, in *Product-Focused Software Process Improvement*. 2006, Springer: Amsterdam, Netherlands. p. 290-304.
17. Mahanti, A., *Challenges in enterprise adoption of agile methods - A survey*. Journal of Computing and Information Technology, 2004. **14**(3): p. 197-206.
18. Cockburn, A., *Agile software development: The Cooperative game*. 2006, Boston, USA: Addison-Wesley Professional.
19. Cohn, M. and D. Ford, *Introducing an agile process to an organization.* Computer, 2003. **36**(6): p. 74-78.
20. Vogel, D., *Agile Methods: Most are not ready for prime time in medical device software design and development*, in *DesignFax Online*. 2006. p. 1-6.

21. Conboy, K., *Agility from first principles: reconstructing the concept of agility in information systems development.* Information Systems Research, 2009. **20**(3): p. 329-354.
22. Schwaber, K. and M. Beedle, *Agile software development with Scrum*. 2001, Upper Saddle River, New Jersey, USA: Prentice Hall.
23. Beck, K. and C. Andres, *Extreme programming explained: Embrace change*. 2004, Boston, USA: Addison-Wesley.
24. Jakobsen, C. and K. Johnson. *Mature agile with a twist of CMMI*. in *Proceedings of the Agile Conference*. 2008. Toronto, Canada: IEEE Computer Society.
25. Lee, G. and W. Xia, *Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility.* Mis Quarterly, 2010. **34**(1): p. 87-114.
26. Chow, T. and D.-B. Cao, *A survey study of critical success factors in agile software projects.* Journal of Systems and Software, 2008. **81**(6): p. 961-971.
27. Cockburn, A. and J. Highsmith, *Agile software development, the people factor*. Computer, 2001. **34**(11): p. 131-133.
28. Rumpe, B. and A. Schröder, *Quantitative survey on extreme programming projects.* arXiv preprint arXiv:1409.6599, 2014.
29. Hansen, M., N. Nohria, and T. Tierney, *What's your strategy for managing knowledge?* Harvard business review, 1999. **77**(2): p. 106-116.
30. Chau, T. and F. Maurer, *Knowledge sharing in agile software teams*, in *Logic versus approximation*, W. Lenski, Editor. 2004, Springer. p. 173-183.
31. Baker, S.W. *Formalizing agility: An agile organization's journey toward CMMI accreditation*. in *Proceedings of the Agile Development Conference*. 2005. Denver, CO, USA: IEEE Computer Society.
32. Theunissen, W.H., D.G. Kourie, and B.W. Watson. *Standards and agile software development*. in *Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology*. 2003. Johannesburg: South African Institute for Computer Scientists and Information Technologists, Republic of South Africa.
33. Cohn, M., *User stories applied: For agile software development*. 2004: Addison-Wesley Professional.
34. McAvoy, J. and T. Butler, *A failure to learn by software developers: Inhibiting the adoption of an Agile software development methodology.* Journal of Information Technology Case and Application Research, 2009. **11**(1): p. 23-46.
35. Galal-Edeen, G., A. Riad, and M. Seyam. *Agility versus discipline: Is reconciliation possible?* in *Proceedings of the International Conference on Computer Engineering & Systems, ICCES*. 2007. Cairo, Egypt: IEEE.
36. Boehm, B. and R. Turner, *People factors in software management: Lessons from comparing agile and plan-driven methods.* Crosstalk - The Journal of Defense Software Engineering, 2003(December): p. 4-8.
37. Standish Group. *2015 CHAOS Report*. 2015 [cited 2016 2nd January]; Available from: https://www.standishgroup.com/store/services/chaos-report-2015-blue-pm2go-membership.html

38. Hastie, S. and S. Wojewoda. *Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch*. 2015 [cited 2016 2nd January]; Available from: http://www.infoq.com/articles/standish-chaos-2015.
39. Bryman, A., *Social Research Methods*. Third Edition ed. 2008, Oxford: Oxford University Press.
40. Schutz, A., *Phenomenology of the Social World*. 1967, Evanston, Illinois: Northwestern University Press.
41. Berger, P.L. and T. Luckmann, *The Social Construction of Reality. A Treatise in the Sociology of Knowledge*. 1966, New York: Doubleday.
42. Laursen, M. and P. Svejvig, *Taking stock of project value creation: A structured literature review with future directions for research and practice.* International Journal of Project Management, 2016. **34**(4): p. 736–747.
43. Turner, R., et al., *Perspectives on Projects*. 2010, London and New York: Routledge.
44. Thomke, S. and T. Fujimoto, *The effect of "front-loading" problem-solving on product development performance.* Journal of Product Innovation Management, 2000. **17**(2): p. 128-142.