On the Benefits/Limitations of Agile Software Development: An Interview Study with Brazilian Companies

Fernando Kamei*, Gustavo Pinto†, Bruno Cartaxo‡∓, Alexandre Vasconcelos‡
*IFAL, Brazil †UFPA, Brazil ‡UFPE, Brazil ∓IFPE, Brazil
fernando.kenji@ifal.edu.br,gpinto@ufpa.br,{bfsc,amlv}@cin.ufpe.br

ABSTRACT

Context: For more than 15 years, Agile Software Development (ASD) has been used to improve software development, process, and quality. However, there are scenarios where the effectiveness of these methods and practices has not been rigorously evaluated.
Objective: Understand the benefits and limitations related to these methods and practices in a particular context: two software companies based on Pernambuco's Technology Park, Brazil.
Method: In this paper, we conducted 22 semi-structured interviews to understand the benefits and limitations of ASD in an industrial context. The data were extracted using open coding and analyzed through qualitative techniques.
Results: Our preliminary analysis identified a core of 28 benefits and 20 limitations with the usage of ASD. As for benefits, we found that facilitates project monitoring and tracking as well as the interaction and collaboration. As for limitations, we found that it difficulty working with user stories and to work with large teams.
Conclusion: This study serves as a practical guide for software companies interested in adopting and improving the use of ASD.

KEYWORDS

Agile Software Development, Agile Practices, Benefits, Limitations, Software Industry, Interviews

ACM Reference format:
Fernando Kamei*, Gustavo Pinto†, Bruno Cartaxo‡∓, Alexandre Vasconcelos‡. 2017. On the Benefits/Limitations of Agile Software Development: An Interview Study with Brazilian Companies. In *Proceedings of EASE'17, Karlskrona, Sweden, June 15-16, 2017,* 6 pages.
DOI: http://dx.doi.org/10.1145/3084226.3084278

1 INTRODUCTION

The fast growth of the global software industry, aligned with the increasing demand for robust methods to deal with existing software, fostered the quest for excellence and continuous improvement of software quality. Along the last years, many software development practices have been proposed, implemented, and evaluated in the industry [7, 11, 15]. Among the notable solutions, Agile Software Development (ASD) stands as a solid, low effort, and useful practice [2, 3, 21]. There is also evidence that ASD is invaluable to reduce the failure rate of software development practice [10].

EASE'17, Karlskrona, Sweden

DOI: http://dx.doi.org/10.1145/3084226.3084278

However, some researchers argue that there is a lack of rigorous empirical studies aimed at evaluating Agile methods and practices. Therefore, such studies are necessary to assess their effectiveness. As a result, little is known about how these methods are employed in practice. Moreover, most of the studies are exploratory in nature [10], therefore questions such as "what are their benefits?", and "what are their limitations?", are not easily answered. This happens because the context of which the software company is deployed (*e.g.,* the number of developers in a team or how they are distributed) does matter [1, 4]. Due to this complex scenario, most related studies do not cover the whole spectrum of context and usage of Agile methods.

In this study, we aimed to reduce this gap by understanding the benefits and limitations related to Agile methods in a specific industrial context: software companies based on the Porto Digital Technology Park[1], in Recife, Brazil. Porto Digital is among the largest technology parks in Brazil, hosting over two hundred highly innovative software companies. More specifically, the research questions we are trying to answer are:

- **RQ1:** What are the benefits of using ASD?
- **RQ2:** What are the limitations of using ASD?

By definition, ASD is a way of working in an agile manner (according to Agile Principles, (*e.g.,* Welcome changing requirements, and Self-organization team) using Agile Methods (*e.g.,* Scrum, XP, and Kanban) or, more specifically, agile Practices (*e.g.,* Pair Programming, Daily Meetings, and Collective Ownership) [9].

To answer these RQs, we employed qualitative research methods with semi-structured interviews. We found several benefits and limitations in this particular context. Our results not only reinforce some of the already known findings of Agile Methods in practice but also shed some light on areas that deserve further investigation. We present a detailed research context information (2). Next, we explain about how the interviews were carried out, and our criteria of extraction and analysis (3). Then, we present the findings regarding the benefits and limitations (4), and a discussion about these (5). Next, we discussed some related works (6). And finally, we present our conclusions (7).

2 CONTEXT DESCRIPTION

In this paper, we argue that context is relevant for assessing the effectiveness of Agile methods in software development [4]. We based our study on the context of two software companies: ALPHA and BETA[2].

[1] http://www.portodigital.org/
[2] We refer to them as "ALPHA" and "BETA" to respect their confidentiality

We collected data from three software projects under development. One from ALPHA and two from BETA, respectively named as A1, B1, and B2. These software companies signed a Term of Authorization and Commitment, which granted researchers access to participants.

ALPHA is a small company created in 2005 which deploys customized software to Brazilian government agencies. It is certified as MPS.BR level G (which is equivalent to CMMI level 1) and has about 60 employees. They employ Scrum as the main Agile method but also employs some XP practices, such as Pair Programming, Real Customer Involvement, Co-location Team, and TDD. Each Sprint lasts two weeks.

Project A1 was under development approximately for one year and employ the technologies Java for a web, JSF, and the Demoiselle Framework. This project aimed to development a financial control system to track and monitor expenses and government revenues. We interviewed 7 out of the 9 team members, who had on average 4.5 years of experience with software development and 2.5 years with Agile methods. All interviewees considered this project to be of medium level of complexity. We summarized the characteristics of each interviewee in Table 1.

Table 1: Characteristics of ALFA's interviewees: Project A1

Role	Age	Dev. exp.	Agile exp.
Software Engineer / Scrum Master	28	4 years	1.5 year
Project Manager	27	6 years	2.5 years
System Analyst / PO	26	5 years	3 years
Test Engineer	22	3 years	1.5 year
Software Engineer	32	5 years	4 years
IT Manager	26	6 years	3 years
Software Engineer	27	2 years	1.5 year

BETA is a medium-sized technology company created in 1996 with more than 700 employees distributed in three Brazilian offices. It is CMMI level 3 certified and delivers products and services to several business areas using Scrum, XP, and Lean Software Development. We selected two projects that use Scrum and some XP practices, such as Pair Programming, TDD, and Co-location team.

Project B1 has been under development for over six months using the C and Java language. This project aimed to develop protocols and network management systems without a GUI. Each Sprint lasts three weeks. The customer could participate at any time during the Sprints. We interviewed 7 out of 9 team members, who had on average five years of experience in software development and two years with Agile methods. All interviewees considered this project to be of high level of complexity. We summarized the characteristics of each interviewee in Table 2.

Project B2 was under development over three years using Java to develop SaaS applications for a telecommunications company. It used Scrum from the very beginning. The customer could also participate at any time during the Sprint using instant messaging tools or email. Two team members played the Product Owner (PO) role. One was a software engineer at the BETA company and the other one was a software engineer at the customer company. We did not interview the latter. We interviewed 8 out of 9 members, who had on average seven years of experience with software development and three years of experience with Agile methods. All interviewees considered this project to be of high level of complexity. We summarized the characteristics of each interviewee in Table 3.

Table 2: Characteristics of BETA's interviewees: Project B1

Role	Age	Dev. exp.	Agile exp.
Software Engineer	25	2 years	1 year
Software Engineer	27	3 years	2 years
Team leader / Scrum Master	34	13 years	2 years
Software Engineer	30	7 years	2 years
Software Engineer	30	2 years	2 years
Software Engineer	29	4 years	1 year
Software Engineer	23	3 years	3 years

Table 3: Characteristics of BETA's interviewees: Project B2

Role	Age	Dev. exp.	Agile exp.
Software Engineer	27	4 years	4 years
Team leader / Scrum Master	30	9 years	4 years
Test Engineer / Product Owner	28	7 years	2 years
Software Engineer	23	3 years	2 years
Software Engineer	25	2 years	2 years
Software Engineer	38	18 years	3 years
Test Engineer	29	7 years	3 years
Project Manager / Scrum Master	28	8 years	6 years

3 INTERVIEWS

We used interviews as our data collection procedure, conducted at the companies' headquarters in October 2014. According to Merriam [17], interviews are effective to elicit information about things that cannot be observed. We used semi-structured interviews with open-ended questions because this approach gathers richer responses when compared to structured interviews. These were conducted in Portuguese since it was the main language of the interviewer and interviewees.

Interviewees: we contacted each participant in advance, and each interview occurred in a private meeting room. Interviewees accepted voluntarily to participate in the research and had to agree with the Informed Consent Form, which guarantees the confidentiality of the data provided, the anonymity of the participants and the right to withdraw from the research at any moment.

Interviews: we conducted 22 semi-structured interviews. The interview had six parts:

(1) We explained the purpose of the study;
(2) We asked questions regarding the participants' background and experience;

(3) We asked questions to understand how participants deal with agile practices;
(4–5) We collected the interviewees' impressions on the (4) benefits and (5) limitations of the use of ASD;
(6) We ended the interview asking whether the participants had additional comments that were not covered by the previous questions.

We collected general information about each project during the first interview, such as team size, the context of the project, the programming languages used, Agile methods, and practices used.

Data Extraction: we recorded all interviews, totaling 23 hours and 19 minutes of audio time. The first author transcribed them, and a form in MS Excel™ was used to guide data extraction.

Data Analysis and Synthesis: we used qualitative coding, similar to *open coding* techniques to identify factors. We used a post-formed code, so we labeled portions of text without any previous pre-formed code. Figure 1 shows two examples of the open coding process used in two interview transcripts. The categories and subset that emerged from the data were analyzed, compared and re-analyzed in parallel, using an approach similar to *axial coding*. Figure 2 shows an example of a category using of axial coding.

Interview transcript: "*Having the task board in the room, it's easy to see how to project is progressing, if it is late or on schedule.*" [A1_4]

Code: Visualize the progress

Interview transcript: "*I think the task board is pretty good, because you just look at the task board and you can see instantly what each member is doing. […] You just need to raise your head and you already see the project status.*" [B2_2]

Code: See what each member is doing

Figure 1: Open coding process used in the interview transcript.

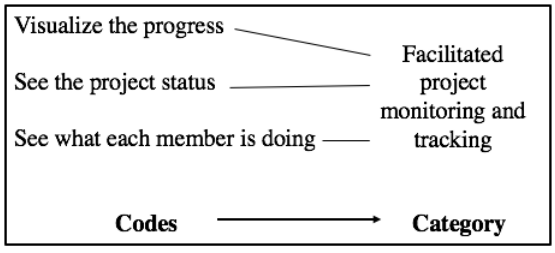

Figure 2: Example of how the category emerged from the initial codes.

To avoid interpretation bias, after the codification phase (Figure 1), we sent our interpretation and each transcript to the interviewees and ask them to highlight any misinterpretation. No problems were raised by the interviewees.

4 RESULTS

When analyzing the interviews, we categorized 28 benefits and 20 limitations. Due to space constraints, we provide discussions on the eight most common ones. When discussing the main benefits (4.1) and limitations (4.2), we correlate them to agile practices, events or principles, quoting opinions from the interviews. Among similar opinions, we chose to only quote the one we considered the most representative for each case.

4.1 What are the benefits of using ASD?

In this section, we present a discussion on some of the benefits found. Table 4 presents a subset of the most common occurrences. The complete list of all benefits and relation to ASD can be found at the companion website: https://goo.gl/qrHo3N.

Table 4: Summary of the benefits. The column # shows the total number of occurrences of a given category.

Benefits	A1	B1	B2	#
Facilitated project monitoring and tracking	×	×	×	21
Facilitated interaction and collaboration	×	×	×	18
Improved communication	×	×	×	16
Improved project understanding	×	×	×	16
Fosters knowledge sharing within the team members	×	×	×	15
Quick/Frequent feedback	×	×	×	14
Facilitated problem-solving	×	×	×	13
Improved quality	×	×	×	13
Problems are discovered/solved earlier	×	×	×	13
Improved software development process	×	×	×	8
Increased commitment	×	×	×	8
Reduced the complexity	×	×	×	7
More agreeable work environment	×		×	6
Increased team's autonomy	×	×	×	6
Reduced amount of defects or errors	×	×	×	6
Improved estimation of software project effort	×	×	×	6
Improved prioritization of requirements		×	×	5
Increased customer satisfaction	×		×	4
Reduced rework	×	×	×	4
Reduced external interference	×		×	3

Facilitated project monitoring and tracking. Many agile practices/events are useful to understand daily the health of the project. For example, the *Daily Meetings* possibility the team members know what each member is doing, and what is done, as an interviewee stated: "*The daily meeting is valid because you follow up and you can get a general sense of the progress*". The use of *Task board* was also seen as beneficial, because anyone in the team can easily see and

understand how well the team is doing during the Sprint, and if it is necessary to take special attention to some task. Recent literature also supports this benefit [1, 8, 10, 22].

Facilitated interaction and collaboration. A subset of benefits are related to this, as interaction and collaboration among the team, and a greater interaction between customers and team members. The core practice was *Co-location Team*, as one test engineer stated *"As we sit together [co-located], the interaction is very close"* and *Daily Meeting* as an interviewee pointed out *"During the daily meetings I can talk about my difficulties, and someone can help me"*. Recent literature also supports this benefit [10, 12, 14, 19].

Improved communication. A subset of benefits are related to this, as communication between team members, and among customer and team members. The main practices were *Daily Meetings* and *Co-location Team*. One interviewee said: *"[Daily meetings] are important because we can communicate more, creating a strong relationship. Thus, we are always communicating"*. Recent literature also supports these benefits [1, 8, 10, 19, 22]. On the other hand, there is evidence showing no positive impact of certain agile methods/practices from the perspective of customer satisfaction [5].

Improved project understanding. Scrum events as *Sprint Planning* and *Review*, and principle of *Real Customer Involvement* were important for this category. In total, we found two subsets: general improvement and help the understanding of User Stories. As a participant stated: *"It serves [Sprint Planning 2] to validate something we did not understand, or something misinterpreted"*. Recent literature also supports this benefit [7, 20].

Fosters knowledge sharing within the team members. Some ASD practices were mentioned to support this category, such as *Pair Programming*, *Sprint Planning*, *Daily Meetings*, *Co-location team*, *Sprint Review*, and *Task board*. According to a software engineer, *"You see your pair doing something [coding], and you think 'I had never thought about that', or help you to use the IDE with new shortcuts. Moreover, you can help your pair if s/he misses something"*. Recent literature also supports this benefit [1, 8, 10, 19].

Quick/Frequent feedback. Some ASD practices and principles were mentioned to support this category, mainly *Iterative and incremental development* and *Sprint Review*. One interviewee pointed out: *"I think that delivering software in small parts is very important, because we have a quick feedback from customer"*. Recent literature also supports this benefit [10, 19, 22].

Facilitated problem-solving. Some ASD practices were mentioned to support this category, as a *Pair Programming* and Scrum Master, as one developer experienced: *"when we do not know how to solve a problem, we contact the Scrum Master. S/he usually can help us"*. Recent literature also supports this benefit [16].

Improved quality .This benefit was perceived mainly concerning the code and the project in general. The main practices to support this benefit were *Pair Programming* and *Sprint Review*. One interviewee pointed out: *"[...] when we are working in a pair, we always do a code review"*.

4.2 What are the limitations of using ASD?

In this section, we present a discussion on some of the limitations found listed in Table 5. The relation of each limitations to ASD can be found at the companion website: https://goo.gl/3Ogyil.

Table 5: Summary of the limitations. The column # shows the total number of occurrences of a given category.

Limitations	A1	B1	B2	#
Difficulty working with User Stories	×	×	×	11
Difficulty working with large teams	×	×	×	9
Increased specialization of team members	×	×	×	9
Interference of Product Owner with technical skills	×	×	×	7
Increased time of activities	×	×	×	6
Difficulty working with a very closed person	×	×	×	6
Difficulty applying/adapting the method/practice	×	×	×	5
Difficulty showing progress in projects that do not have GUI		×		3
Difficulty working with non-agile people	×			3
Requires maturity, commitment, and pro-activeness from team members	×	×		3
Difficulty concentrating in co-location teams	×	×		2
Difficulty identifying individual contributions	×			2
Difficulty working with open scope contracts	×			2
Increased costs of the project	×		×	2
Little documentation			×	2
Difficulty managing dependent requirements			×	1
Difficulty monitoring distributed projects	×			1
Formalism on meetings can inhibit good communication	×			1
Increased pressure for delivery of the work			×	1
Little focus on architecture development			×	1

Difficulty working with User Stories. A subset of limitations are related to this, as difficult to estimate, to split into tasks, to be used in complex projects, and to understand the user stories, as interviewees pointed out: *"the use of story points does not work well because it is subjective. [...] I believe that estimate the effort in hours is more accurate"* and *"It is difficult to estimate in points when the technology is unknown."*. Recent literature also supports this limitation [6, 10].

Difficulty working with large teams. Although one of the principles of Agile Methods is to work with small teams, this limitation is not new in literature, and was found in many interviews mainly due to the events of *Daily Meetings* and *Sprint Planning*, as an interviewee stated: *"You lose focus when you have a Sprint Planning*

or Daily Meeting with a large team". Recent literature also supports this limitation [1, 10, 22].

Increased specialization of team members. Several agile principles emphasize that teams working on the same project should work together at the same place (*Co-location team*). However, they faced problems when testers and developers worked together, which happened during *Daily Meetings*, *Sprint Planning*, and *Sprint Review*, as an interviewee stated: *"Some discussions are exclusive to testers. Some discussions are exclusively to developers. Eventually, it is unnecessary for me, because I could not help"*.

Interference of Product Owner with technical skills. Agile methods emphasize constant communication between customers and the development team. However, we found that the Product Owner can become a burden, as an interviewee stated: *"the Product Owner had technical skills, but do not understand the core of the source code in greater details. He wants to push up many user stories"*.

Increased time of activities. This limitation is a subset on the increase on development time, on the time of meetings, and management overhead, mainly caused by ASD events, such as *Daily meetings*, *Sprint Planning and Review*, and *TDD*. One interviewee pointed out: *"TDD is quite good, but you are always evolving, resulting in double of time."*

Difficulty working with a very closed person. This limitation occurs because the communication and collaboration among stakeholders may involve dealing with different personality traits, as one interviewee stated: *"[...] We have a highly skilled team member, but he is a very closed person. So, when I need some help, I do not feel comfortable to talk with him"*. This limitation highlights that non-technical skill are also valuable for software developers. Recent literature also supports this limitation [1, 8, 10].

Difficulty applying/adapting the method/practice. A subset of limitations are related to this, as *difficult to implement TDD technique*, and *difficult to use Pair Programming*. Although ASD embraces changes, this was a problem for one interviewee: *"[...] sometimes the change is radical, and practically everything has changed. Some changes are quite difficult to handle."*.

Difficulty showing progress in projects that do not have GUI. This limitation occurs due to the particularity of B1 project during the Sprint Review, as a team leader stated: *"[...] it is very difficult to show something functional to the PO (Product Owner) due to the nature of the project"*.

5 DISCUSSION

In this section, we revisit the main findings of this study (Section 5.1) and discuss important threats to validity (Section 5.2).

5.1 Revisiting findings

Our results suggest that the benefits and limitations of ASD are not yet fully understood. We observed that one given agile practice could cause both benefits and limitations. For example, although some interviewees agree that ASD can fosters knowledge sharing within the team members, other interviewees suggested that pairing with colleagues can also be a burden. Therefore, practitioners might place additional care when introducing agile practices in their projects. We also observed that practitioners need to have a better understanding not only of the technical details of agile practices, but also of the context in which the project is inserted, and the cultural characteristics of participants — as some limitations were related to personal traits.

Benefits & Limitations. In regards to the benefits, the interviewees perceived the *Sprint Planning* as beneficial (it was related to 14 benefits). This practice was most frequently on the benefit *Fosters knowledge sharing within the team members*. Other practices most commons, related to 8 benefits were: *Daily Meetings*, *Iterative and Incremental Development*, *Real customer involvement* and *Short iterations*. As regarding the limitations, the *Sprint Planning* meeting was the most cited practice. One limitation regarding is that *Requires maturity, commitment, and pro-activeness from team members*, since, to have a successful Sprint, it is important to have a self-management team [13].

Does context matter? In this paper we studied two companies: a small one (60+ employees) and a large one (700+ employees). However, we observed that the developers' characteristics and backgrounds are more relevant to the benefits and limitations found, than the size of the company. This is partially because, although the software companies have different sizes, the size of the teams were the same: they all have nine members.

5.2 Threats to Validity

First, to avoid inhibition during the interviews and thus compromising our data collection, all interviewees had to sign the Informed Consent Form. In this form, we take responsibility in keeping confidential any information provided. Second, to foster diversity, we also collected data from participants playing different roles. We also studied three projects under active development in two distinct Brazilian software companies. However, only one researcher performed the analysis and interpretation of data, which might introduce some bias due to the qualitative nature of this research. To mitigate this bias, we sent the results to our interviewees so that they might indicate whether we misinterpreted some discussions. We observed that some teams use Scrum with variations (e.g., some findings suggested team members discussions during the daily meetings). However, in this study, we do not place emphasis in differentiating Scrum variations, since we are interested in a broader understanding of the benefits and limitations of this practice. Still, in addition to the detailed description of the results found, we also observed that our findings are in line with the recent literature. Finally, to foster replicability, we made available all benefits and limitations found.

6 RELATED WORK

Begel and Nagapan [1] conducted a web survey at Microsoft to identify the perceptions of Agile Methods. They reported 25 benefits and 14 limitations. The findings of our study overlap with this in 13 out of 28 benefits (*e.g., Improved communication*, and *Quick/Frequent feedback*), and 7 out of 20 limitations (e.g. *Difficulty working with a very closed person* and *Little documentation*).

Dybå and Dingsøyr [10] conducted a systematic review targeting Agile methods. They included 36 primary studies. They found 37 benefits and 12 limitations, primarily with the use of eXtreme Programming. On our research, however, all investigated projects

used Scrum, and some XP practices. The findings of our study overlap with this in 20 out of 28 benefits (*e.g., Facilitated project monitoring and tracking*, and *Quick/Frequent feedback*) and 5 out of 20 limitations (*e.g., Increased time of activities* and *Difficulty working with a very closed person*).

Petersen and Wohlin [19] identified the advantages and problems of Agile Methods in a large scale project. As in our study, they conducted 33 semi-structured interviews. They found 14 benefits and 12 limitations. The findings of our study overlap with this study in 12 out of 28 benefits (*e.g., Quick/Frequent feedback*, *Problems are discovered/solved earlier*, and *Improved communication*), and 7 out of 20 limitations (*e.g., Increased time of activities*, and *Increased pressure for delivery of the work*).

In a recent study, Solinski and Petersen [22] used the extended hierarchical voting analysis framework with 45 agile practitioners to investigate benefit and limitation prioritization. The findings of our study overlap with this study in 10 out of 28 benefits (*e.g., Improved software development process*, *Improved planning and management*, and *Increased efficiency of responses to changing requirements*), and 5 out of 20 limitations (*e.g., Difficulty working with large teams*, *Increased pressure for delivery of the work*, and *Increased specialization of team members*).

Melo *et al.* [8] presented an overview of the evolution of the agile movement in Brazil. Although their work is both quantitative and qualitative, the findings of our study overlap with this study in 8 out of 28 benefits (*e.g., Increased customer satisfaction*, and *Increased team motivation*), and 5 out of 20 limitations (*e.g., Requires maturity, commitment, and pro-activeness from team members*, and *Difficulty working with a very closed person*).

The "State of Agile" is an annual known research survey conducted by VersionOne. The 11th edition [18] was conducted in 2016. The findings of our study overlap with this study in 8 out of 28 benefits (*e.g., Increased team productivity*, and *Improved quality*), and 5 out of 20 limitations (*e.g., Difficulty working with large teams*, and *Difficulty applying/adapting the method/practice*).

7 CONCLUSION

In this paper, we analyzed the benefits and limitations of the use of Agile Software Development (ASD) in the context of two Brazilian software companies. Through 22 semi-structured interviews, we created a curated list of 28 benefits and 20 limitations. We found that several benefits are related to (1) improving project monitoring and tracking, (2) improving interaction and collaboration, and (3) fosters sharing knowledge, whereas the limitations are (1) difficulty working with user stories, (2) difficulty working with large teams, and (3) increased specialization of team members. As a result, we believe that practitioners should better understand not only ASD but also the context in which a project emerges, as well as the cultural characteristics of participants.

For future work, we plan to correlate the findings from different software companies and countries to better understand whether the benefits and limitations found can be generalized. We also plan to create an "Agile Practice Impact Model" to better visualize and investigate which agile practice influences a given benefit or limitation.

Acknowledgements We wish to thank anonymous reviewers for helping improve this paper. This research was partially funded by CNPq (406308/2016-0).

REFERENCES

[1] Andrew Begel and Nachiappan Nagappan. 2007. Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study. In *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM '07)*. 255–264.
[2] Barry Boehm. 2002. Get ready for agile methods, with care. *Computer* 35, 1 (2002), 64–69.
[3] Barry Boehm. 2006. A View of 20th and 21st Century Software Engineering. In *Proceedings of the 28th International Conference on Software Engineering (ICSE '06)*. 12–29.
[4] Bruno Cartaxo, Adaulto Almeida, Emanoel Barreiros, Juliana Saraiva, Waldemar Ferreira, and Sérgio Soares. 2015. Mechanisms to characterize context of empirical studies in software engineering. In *Proceedings of the Workshop on Experimental Software Engineering (ESELAW '15)*.
[5] Bruno Cartaxo, Allan Arajo, Antonio Sa Barreto, and Sérgio Soares. 2013. The Impact of Scrum on Customer Satisfaction: An Empirical Study. In *Proceedings of the 27th Brazilian Symposium on Software Engineering (SBES '13)*. 129–136.
[6] Bruno Cartaxo, Gustavo Pinto, Danilo Monteiro Ribeiro, Fernando Kamei, Ronnie E. S. Santos, Sérgio Soares, and Fabio Q. B. Da Silva. 2017. Using Q&A Websites as a Method for Assessing Systematic Reviews. In *Proceedings of the 14th International Conference on Mining Software Repositories (MSR '17)*.
[7] Tsun Chow and Dac-Buu Cao. 2008. A survey study of critical success factors in agile software projects. *Journal of Systems and Software* 81, 6 (Jun. 2008), 961–971.
[8] Claudia de O. Melo, Viviane A. Santos, Eduardo T. Katayama, Hugo Corbucci, Rafael Prikladnicki, Alfredo Goldman, and Fabio Kon. 2013. The evolution of agile software development in Brazil - Education, research, and the state-of-the-practice. *Journal of the Brazilian Computer Society,* 19, 4 (Nov. 2013), 523–552.
[9] Philipp Diebold and Thomas Zehler. 2016. *The Right Degree of Agility in Rich Processes.* Springer International Publishing, Cham, 15–37. DOI: http://dx.doi.org/10.1007/978-3-319-31545-4_2
[10] Tore Dybå and Torgeir Dingsøyr. 2008. Empirical Studies of Agile Software Development: A Systematic Review. *Information and Software Technology* 50, 9-10 (Aug. 2008), 833–859.
[11] Martin Fowler. 1999. *Refactoring: Improving the Design of Existing Code.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
[12] Rashina Hoda, James Noble, and Stuart Marshall. 2010. Balancing Acts: Walking the Agile Tightrope. In *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering (CHASE '10)*. 5–12.
[13] Rashina Hoda, James Noble, and Stuart Marshall. 2010. Organizing Self-organizing Teams. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1 (ICSE '10)*. 285–294.
[14] Emam Hossain, Muhammad Ali Babar, and June Verner. 2009. How Can Agile Practices Minimize Global Software Development Co-ordination Risks?. In *Proceedings of the 16th European Conference on Software Process Improvement (EuroSPI '09)*. Berlin, Heidelberg, 81–92.
[15] Watts S Humphrey. 1995. *A discipline for software engineering.* Addison-Wesley Longman Publishing Co., Inc.
[16] Seiyoung Lee and Hwan-Seung Yong. 2010. Distributed agile: project management in a global environment. *Empirical Software Engineering* 15, 2 (Apr. 2010), 204–217.
[17] Sharan B Merriam and Elizabeth J Tisdell. 2015. *Qualitative research: A guide to design and implementation.* John Wiley & Sons.
[18] Version One. 2016. 11th Annual State of Agile Report. (March 2016). Retrieved April 26, 2017 from https://explore.versionone.com/state-of-agile/versionone-11th-annual-state-of-agile-report-2.
[19] Kai Petersen and Claes Wohlin. 2009. A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *Journal of Systems and Software* 82, 9 (Sep. 2009), 1479–1490.
[20] Minna Pikkarainen, Jukka Haikara, Outi Salo, Pekka Abrahamsson, and Jari Still. 2008. The Impact of Agile Practices on Communication in Software Development. *Empirical Software Engineering* 13, 3 (Jun. 2008), 303–337.
[21] D. J. Reifer. 2002. How good are agile methods? *IEEE Software* 19, 4 (Jul. 2002), 16–18.
[22] Adam Solinski and Kai Petersen. 2016. Prioritizing agile benefits and limitations in relation to practice usage. *Software Quality Journal* 24, 2 (Jun. 2016), 447–482.