e-Informatica Software Engineering Journal, Volume 15, Issue 1, 2021, pages: 47–64, DOI 10.37190/e-Inf210103

Use of Agile Practices in Start-up Companies

Eriks Klotins*, Michael Unterkalmsteiner*, Panagiota Chatzipetrou**, Tony Gorschek***, Rafael Prikladnicki****, Nirnaya Tripathi*****, Leandro Bento Pompermaier******

*Software Engineering Research Lab, Blekinge Institute of Technology
**Department of Informatics, CERIS, Örebro University School of Business
***Software Engineering Research Lab, Blekinge Institute of Technology
****Software Engineering Research Lab, Pontifical Catholic University of Rio Grande do Sul
*****Software Engineering Research Lab, University of Oulu
******Software Engineering Research Lab, Pontifical Catholic University of Rio Grande do Sul

eriks.klotins@bth.se, michael.unterkalmsteiner@bth.se, panagiota.chatzipetrou@oru.se, tony.gorschek@bth.se, rafael.prikladnicki@pucrs.br, Nirnaya.Tripathi@oulu.fi, leandro.pompermaier@pucrs.br

Abstract

Context Software start-ups have shown their ability to develop and launch innovative software products and services. Small, motivated teams and uncertain project scope makes start-ups good candidates for adopting Agile practices.

Objective We explore how start-ups use Agile practices and what effects can be associated with the use of those practices.

Method We use a case survey to analyze 84 start-up cases and 56 Agile practices. We apply statistical methods to test for statistically significant associations between the use of Agile practices, team, and product factors.

Results Our results suggest that development of the backlog, use of version control, code refactoring, and development of user stories are the most frequently reported practices. We identify 22 associations between the use of Agile practices, team, and product factors. The use of Agile practices is associated with effects on source code and overall product quality. A teams' positive or negative attitude towards best engineering practices is a significant indicator for either adoption or rejection of certain Agile practices. To explore the relationships in our findings, we set forth a number of propositions that can be investigated in future research.

Conclusions We conclude that start-ups use Agile practices, however without following any specific methodology. We identify the opportunity for more fine-grained studies into the adoption and effects of individual Agile practices. Start-up practitioners could benefit from Agile practices in terms of better overall quality, tighter control over team performance, and resource utilization.

Keywords: Agile practices, start-up companies

1. Introduction

Software start-ups are important suppliers of innovation, new products, and services. However, engineering of software in start-ups is a complicated endeavor as the start-up context poses challenges to software engineers [1]. As a result of these challenges, most start-ups do not survive the first few years of operation and cease to exist before delivering any value [2, 3].

Uncertainty, changing goals, limited human resources, extreme time, and resource constraints are reported as characteristics of start-ups [1, 4].

To survive in such a context, start-ups use ad hoc engineering practices and attempt to tailor agile methods to their needs. However,

Submitted: 31 August 2020; Revised: 04 December 2020; Accepted: 12 January 2021; Available online: 29 March 2021

scaled-down agile methods could be irrelevant and ignore start-up specific challenges [5, 6].

Giardino et al. [7] suggest that start-ups adopt practices as a response to some problematic situations and do not consider adopting full agile methodologies, e.g., scrum or XP, at least in early stages.

Pantiuchina et al. [8] make a similar observation and argue that start-ups focus more on speed-related practices, e.g., iterations and frequent releases, than quality-related practices, e.g., unit testing and refactoring.

In this study, we explore the use of Agile practices in start-ups. We focus on identifying the associations between certain Agile practices, product, and team factors. We aim to understand what positive, and potentially adverse effects can be associated with the use of specific practices. We use our results to formulate propositions for further exploration.

We use a case survey to collect data from 84 start-up cases [9]. We use statistical methods to analyze 11,088 data points and identify associations between the use of Agile practices and respondents' estimates on various team and product factors.

We identify 20 statistically significant associations pointing towards potential causes and effects of using Agile practices. We identify that the use of automated tests and continuous integration is associated with positive attitudes towards following best practice. However, the use of planning and control practices are more associated with negative attitudes towards following the best practices.

The rest of this paper is structured as follows. In Section 2, we discuss related work. Section 3 covers the research methodology, data collection, and our approach to data analysis. Section 4 presents the results. We answer our research questions and discuss the implications for research and practice in Section 5. Section 6 concludes the paper.

2. Background and related work

2.1. Software start-ups

Software start-ups are small companies created for developing and bringing an innovative software intensive product or service to the market, and to benefit from economies of scale.

Start-up companies rely on external funding to support their endeavors. In 2015 alone, start-up companies have received investments of 429 billion USD in the US and Europe alone [10, 11]. With an optimistic start-up failure rate of 75% that constitutes of 322 billion USD of capital potentially wasted on building unsuccessful products.

Earlier studies show that product engineering challenges and inadequacies in applied engineering practices could be linked to start-up failures [1, 12]. To what extent software engineering practices are responsible or linked to the success rate is very hard to judge. However, if improved software engineering practices could increase the likelihood of success by only a few percent, it would yield a significant impact on capital return.

Some authors, e.g., Sutton [13] and Giardino [3], point out the unique challenges in start-ups, such as high risk, uncertainty, lack of resources, rapid evolution, immature teams, and time pressure among other factors. At the same time, start-ups are flexible to adopt new engineering practices, and reactive to keep up with emerging technologies and markets [7]. However, our earlier study [12] analyzing the amount of empirical evidence supporting the uniqueness of start-ups found that most start-up characteristics are based on anecdotal evidence. Thus, there could be a negligible difference between start-ups and other organizations launching new software-intensive products on the market in terms of software engineering.

2.2. Agile practices

Agile software engineering practices originate from the Agile manifesto, proposing a shift from heavyweight, plan-driven engineering towards more lightweight, customer-oriented, and flexible methodologies [14]. Agile methodologies, such as Scrum and XP, prescribe specific sets of Agile practices [15, 16]. However, in practice, by-the-book methodologies are often tailored with additional practices to address specific concerns [17, 18]. Thus, we focus our study on what

practices start-ups use, without considering any specific agile methodology.

Small organizations have successfully adopted Agile practices for projects where requirements are uncertain and expected to change [19, 20]. In theory, Agile practices could be perfect for software start-ups [6]. However, successful adoption of Agile practices requires highly skilled teams and support throughout the organization [19, 21].

Earlier work on software engineering practices in start-ups suggests that start-ups initially rely on an ad hoc approach to engineering and adopt agile principles incrementally when the need for more systematic practice arises. The shift is often motivated by excessive technical debt, hindering quality, and lack of control over the engineering process [7].

The motivations for adopting agile practices in start-ups include accelerated product delivery, ability to manage changing priorities, and increased team productivity. Practices concerning team collaboration such as open work areas, use of task boards, and a prioritized backlog are reported as the most widely used [22]. Souza et al. [23] reports that start-ups primarily adopt practices that provide immediate benefits and help to accelerate time-to-market.

We explore the associations between 56 Agile practices, product, and team factors. We use a list and descriptions of Agile practices compiled by Agile Alliance, a non-profit community promoting agile principles [24]. To our best knowledge, their website contains the most comprehensive list of Agile practices to date.

In this study, we consider the following practices whose definitions can be found at the Agile Alliance's website [24]: Card, Conversation, Confirmation (3C's), Acceptance tests, Acceptance Test-Driven Development (ATDD), Automated build, Backlog, Backlog grooming, Behavior Driven Development, Burndown chart, Collective ownership, Continuous deployment, Continuous integration, Class Responsibility Collaborator (CRC) Card cards, Daily meeting, Definition of Done, Definition of Ready, Exploratory testing, Facilitation, Frequent releases, Given--When-Then, Heartbeat retrospective, Incremental development, INVEST, Iterations, Iterative development, Kanban board, Lead time, Mock objects, Niko-Niko, Pair Programming, Personas, Planning poker, Point estimates, Project charters, Quick design session, Refactoring, Relative estimation, Role-Feature-Reason, Rules of simplicity, Scrum of Scrums, Sign up for tasks, Simple design, Story mapping, Story splitting, Sustainable Pace, Task board, Team, Team room, Test-driven development, Three Questions, Timebox, Ubiquitous language, Unit tests, Usability testing, User stories, Velocity, and Version control.

In this paper, we follow Agile Alliance naming of the practices. Some of the terms describing practices can also refer to artifacts, e.g., acceptance tests. When we use such a term, we mean the practice of creating and utilizing acceptance tests.

2.3. Effects of using Agile practices

The use of Agile practices is associated with increased product quality and fewer defects compared to plan-driven approaches [25, 26]. We analyze the associations between the use of Agile practices, product documentation, software architecture, quality of the source code, tests, and the overall product quality. In this paper, we adopt the product view on software quality, recognizing the relationship between internal product characteristics and quality of use [27].

Product documentation comprises of written requirements, architecture documentation, and test cases. Deficiencies in such artifacts are associated with hindered knowledge distribution in the team and with adverse effects on further development and maintenance of the product [28]. Note that we analyze if documentation artifacts are understandable and useful without implying any specific format.

Even though the Agile manifesto emphasizes working software over comprehensive documentation, some documentation is essential [14]. For example, user stories are one of the key agile tools to document requirements [29]. System metaphor is useful to communicate the logical structure of the software to all stakeholders [30]. The use of

automated testing in continuous integration and deployment pipelines require formally defined tests [31].

Software architecture denotes how different components, modules, and technologies are combined to compose the product. Symptoms such as outdated components, a need for workarounds and patches point towards deficiencies in the software architecture and the lack of attention to refactoring [32, 33].

Source code quality is determined using coding standards and refactoring practices [34, 35]. Degrading architecture and poorly organized source code is associated with increased software complexity, difficult maintenance, and product quality issues down the road [28].

We analyze the quality (or lack thereof) of automated test scripts, removing the need to perform manual regression testing on every release of the product. The effort of manual regression testing grows exponentially with the number of features, slowing down release cycles and making defect detection a time-consuming and tedious task [28].

We also examine the associations between product quality and the use of Agile practices. With product quality, we understand nonfunctional aspects of the product, such as performance, scalability, maintainability, security, robustness, and the ability to capture any defects before the product is released to customers [28].

Good communication, teamwork, adequate skills, and a positive attitude towards the following best practices are recognized as essential team factors for project success [19]. Agile software engineering practices aim to facilitate communication, empower individuals, and improve teamwork [36]. We analyze the associations between team characteristics and the use of specific Agile practices.

Attitudes determine the level of apathy or interest in adopting and following the best engineering practices. Skills characterize to what extent individual members of a start-up team possess relevant engineering skills and knowledge. Communication captures to what extent the team can communicate and coordinate the engineering work. Giardino et al. [7] identify the team as the catalyst for product development in start-ups. Sufficient skills, positive attitudes, and efficient communication are essential for rapid product development in both agile and start-up contexts [7, 19].

Pragmatism characterizes to what extent a team can handle trade-offs between investing in perfected engineering solutions and time-to--market. Agile practices advocate for frequent releases and good-enough solutions [15]. Such practices help to validate the product features early and gather additional feedback from customers [12]. On the other hand, quick product releases need to be accompanied by frequent refactoring and unit tests to manage technical debt and keep regression defects under control [19]. Start-ups often overlook such corrective practices [7, 12].

Sufficient time and resources for product engineering are essential for project success [19]. We analyze what Agile practices can be associated with better resource estimation and planning in start-ups. Several authors, e.g., Giardino et al. [3] and Sutton [13] identify resource shortage as one of the critical challenges in start-ups. However, we, in our earlier study identify the lack of adequate resources, planning and control practices in early start-ups [9].

We look into respondent estimates on the engineering process in their organizations. Process characterizes to what extent product engineering is hindered by unanticipated changes in organizational priorities, goals, and unsystematic changes in the product itself. Lack of organizational support for agile product engineering contributes to project failures [19]. On the other hand, Agile practices offer some room for adjusting to unclear and changing objectives [20].

Agile methods on a high level attempt to address and promise improvements in all these concerns [36]. However, analyzing the effects of applying the whole methodology on a large number of factors does not help to pinpoint specific practices for specific challenges. We aim to establish a fine-grained view on the use and effects of individual practices.

3. Research methodology

3.1. Research aim

We aim to explore how software start-ups use Agile practices and what positive and negative effects can be associated with specific practices.

3.2. Research questions

To guide our study, we define the following research questions (RQ):
RQ1: How are Agile practices used in start-ups?
Rationale: With this question, we identify what Agile practices and in what combinations start-ups use.
RQ2: What are the associations between specific Agile practices and product factors?
Rationale: With this question, we explore the associations between specific Agile practices, quality of documentation, architecture, source code, testing, and overall product quality.
RQ3: What are the associations between specific Agile practices and team factors?
Rationale: With this question, we explore the associations between specific Agile practices, attitudes towards following best engineering practices, pragmatism, communication, skills, resources, engineer process, and teams' productivity.

3.3. Data collection

We used a case survey method to collect primary data from start-up companies [9, 37].

The case survey method is based on a questionnaire and is a compromise between a traditional case study and a regular survey [38]. We have designed the questionnaire to collect practitioners' experiences in specific start-up cases.

During the questionnaire design phase, we conducted multiple internal and external reviews to ensure that all questions are relevant, clear and that we receive meaningful answers. First, the questions were reviewed in multiple rounds by the first three authors of this paper to refine the scope of the survey and question formulations. Then, with the help of other researchers from the Software Start-up Research Network[1], we conducted a workshop to gain external input on the questionnaire. A total of 10 researchers participated and provided their input.

Finally, we piloted the questionnaire with four practitioners from different start-ups. During the pilot, respondents filled in the questionnaire while discussing questions, their answers, and any issues with with the first author of this paper.

As a result of these reviews, we improved the question formulations and removed some irrelevant questions. The finalized questionnaire contains 85 questions in 10 sections. The questionnaire captures 285 variables from each start-up case.

We use a list of 56 Agile practices to capture the respondent's answers on what practices they use in their companies, as described in Section 2.2. The answers are captured in a binary use or not use format. In addition to specific practices, we offer an "I do not know" and "other" option to accommodate for the lack of respondents knowledge and to discover other, unlisted practices. We rely on the respondents best judgment to gauge whether the extent of using a practice in their start-ups qualifies as an application of the practice or not.

We use 45 other questions to capture respondents evaluations of product and team-related statements, such as:

- Initial product/service architecture has become outdated;
- Communication and collaboration within the development team regarding the product/service architecture is insufficient;
- Incremental changes to the product/service are unsystematic and degrades the architecture;
- Quick delivery of functionality is considered more important than good code.

The questions capture the respondents' agreement with a statement characterizing a factor on a Likert scale: not at all (1), a little (2), somewhat (3), very much (4). The values indicate the degree of agreement with a statement. Statements are formulated consistently in a way that

[1]The Software Start-up Research Network, https://softwarestartups.org/

lower values indicate less, and higher values indicate more agreement with the statement.

The questionnaire is designed to be filled in by one person and we analyze one response per start-up. To control this, we collect the contact information of the respondent and the title of their company. In addition to questions about software engineering, the questionnaire contains questions inquiring about the respondents background, and engineering context in the start-up. The full questionnaire is available as supplemental material on-line[2].

The data collection occurred between December 1, 2016, and June 15, 2017. The survey was promoted through personal contacts, by attending industry events, and with posts on social media websites. The survey was promoted as "help us to understand what engineering practices work and does not work in start-ups" and targeted for practitioners with an understanding about the engineering parts of their start-up.

We invited other researchers from the Software Start-up Research Network to collaborate on the data collection. This collaboration helped to spread the survey across many geographical locations in Europe, North and South America, and Asia.

3.4. Data analysis methods

To analyze the survey responses, we used several techniques. We started by screening the data and filtering out duplicate cases, responses with few questions answered, or otherwise unusable responses. In the screening, we attempt to be as inclusive as possible and do not remove any cases based on the provided responses.

Overall, we analyzed responses from 84 start-up cases, 132 data points per each case, and 11,088 data points. We use the Chi-Squared test of association to test if the associations between the examined variables are not due to chance. To prevent Type I errors, we used exact tests, specifically, the Monte-Carlo test of statistical significance based on 10,000 sampled tables and assuming ($p < 0.05$) [39].

To examine the strength of associations, we use Cramer's V test. We interpret the test results as suggested by Cohen [40], see Table 1. To explore the specifics of the association, such as which cases are responsible for this association, we performed post hoc testing using adjusted residuals. We consider an adjusted residual significant if the absolute value is above 1.96 ($Adjusted\ residual > 1.96$), as suggested by Agresti [41].

Table 1. Interpretation of Cramer's V test

Cramer's V value	Interpretation
≥ 0.1	Weak association
≥ 0.3	Moderate association
≥ 0.5	Strong association

The adjusted residuals drive our analysis on how different groups of start-ups estimate aspects of technical debt. However, due to the exploratory nature of our study, we do not state any hypotheses upfront and drive our analysis with research questions.

3.5. Validity threats

In this section, we follow the guidelines by Runeson et al. [42] and discuss four types of validity threats and applied countermeasures in the context of our study.

3.5.1. Construct validity

Construct validity concerns whether operational measures represent the studied subject [42]. A potential threat is that the statements we use to capture respondent estimates are not capturing the indented team and product factors.

To address this threat, we organized a series of workshops with other researchers and potential respondents to ensure that the questions are clear to the point and to capture the studied phenomenon.

We triangulate each factor by capturing it by 3–4 different questions in the questionnaire. To avoid biases stemming from respondents precon-

[2]Full questionnaire: http://eriksklotins.lv/files/GCP_questionnaire.pdf

ceived opinions about the effects of agile practices, we separate questions about the use of practices and questions inquiring about team and product factors.

To accommodate for the fact that a respondent may not know the answers to some of the questions, we provide an explicit "I do not know" answer option to all Likert scale questions.

Regarding the use of agile practices, we ask a binary question capturing the use/not use of a practice. Such an approach does not capture the extent, nor other specifics of the application of a practice. This creates a room for a wide interpretation of what entails using the practice. For example, cases having one automated test and cases with an extensive suite of automated tests would be treated the same.

In this study, we aim for breadth, in terms of studied cases and practices, to understand what agile practices are relevant to start-ups and the team and product factors associated with specific practices. Details of the optimal use of the practices is an avenue for further work.

3.5.2. Internal validity

This type of validity threat addresses causal relationships in the study design [42].

In our study, we do not seek to establish causal relationships, thus this type of validity threat is not relevant.

3.5.3. External validity

This type of validity threat concerns to what extent the results could be valid to start-ups outside the study [42]. The study setting for participants was close to real life as possible. That is, the questionnaire was filled in without researcher intervention and in the participant's environment.

The sampling of participants is a concern to external validity. We use convenience sampling to recruit respondents and with the help of other researchers, distributed the survey across several different start-up communities. Demographic information from respondent answers shows that our sample is skewed towards active companies, respondents with little experience in start-ups, young companies, and small development teams of 1–8 engineers. In these aspects, our sample fits the general characteristics of start-ups, see, for example, Giardino et al. [1, 3] and Klotins et al. [5]. However, there is a survivor bias, that is, failed start-ups are underrepresented. Thus, our results reflect state-of-practice in active start-ups.

Another threat to external validity stems from case selection. We marketed the questionnaire to start-ups building software-intensive products. However, due to the broad definition of software start-ups (see Giardino et al. [3]), it is difficult to differentiate between start-ups and small-medium enterprises. We opted to be as inclusive as possible and to discuss relevant demographic information along with our findings.

3.5.4. Conclusion validity

This type of validity threat concerns the possibility of incorrect interpretations arising from flaws in, for example, instrumentation, respondent and researcher personal biases, and external influences [42].

To make sure that respondents interpret the questions in the intended way, we conducted several pilots, workshops and improved the questionnaire afterwards. To minimize the risk of systematic errors, the calculations and the first and the third author performed statistical analysis independently, and the findings were discussed among the authors.

It could be that some respondents may lack the knowledge to fully answer our questions. We mitigate this threat by providing the "I do not know" option to all our questions. We further analyze the respondent demographics and background (see Section 4) to gauge the credibility of their responses. However, we cannot exclude that in some cases the responses are incomplete. As a result, we cannot reliably make conclusions from the absence of information in the responses.

To test if the order of appearance of Agile practices affects practitioner responses, we run a Spearman's rank-order correlation test [43]. We examine a potential relationship between the order of appearance and the frequency chosen by

respondents. The results showed that there is no statistically significant correlation ($p > 0.05$).

4. Results

The majority of the surveyed start-ups (63 out of 84, 75%) are active and have been operating for 1–5 years (58 out of 84, 69%). Start-ups are geographically distributed among Europe (34 out of 84, 40%), South America (41 out of 84, 49%), Asia (7 out of 84, 8%), and North America (2 out of 84, 2%).

Our sample is about equally distributed in terms of the product development phase. We follow the start-up life-cycle model proposed by Crowne [44] and distinguish between inception, stabilization, growth, and maturity phases. In our sample, 16 start-ups have been working on a product but have not yet released it to the market, 24 teams have released the first version and actively develop it further with customer input, 26 start-ups have a stable product and they focus on gaining customer base, and another 16 start-ups have mature products, and they focus on developing variations of their products.

The questionnaire was filled in mostly by start-up founders (64 out of 84, 76%) and engineers employed by start-ups (15 out of 84, 18%). About half of the respondents have specified that their area of expertise is software engineering (49 out of 84, 58%). Others have specified marketing, their respective domains, and business development as their areas of expertise.

The respondents' length of software engineering experience ranges from 6 months to more than 10 years. A large portion of respondents (44 out of 84, 52%) had less than 6 months of experience in working with start-ups at the time when they joined their current start-up.

We provide a complete list of studied cases and their demographical information as supplemental material on-line[3].

The responses on what development type best characterizes the company suggest that most companies, 51 out of 84, 60%, follow agile and

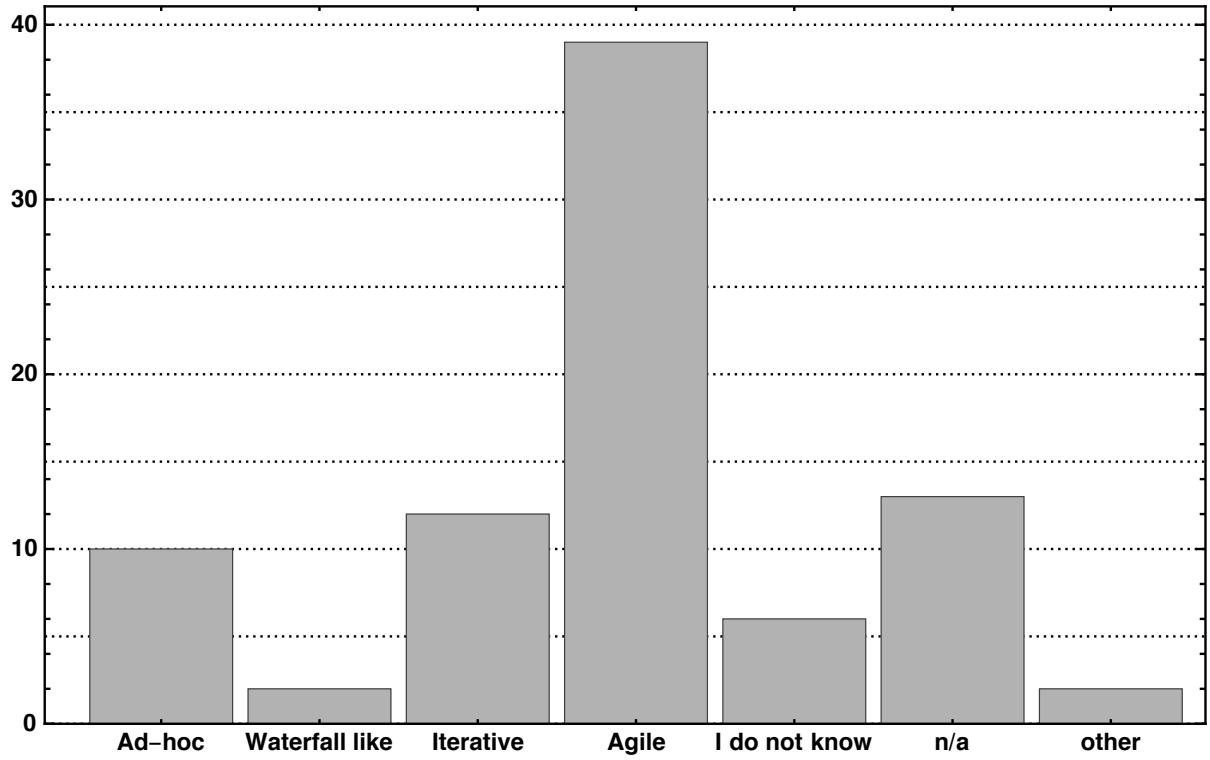

Figure 1. Use of development approaches in the studied cases

[3]The studied cases: http://eriksklotins.lv/files/GCP_demographics.pdf

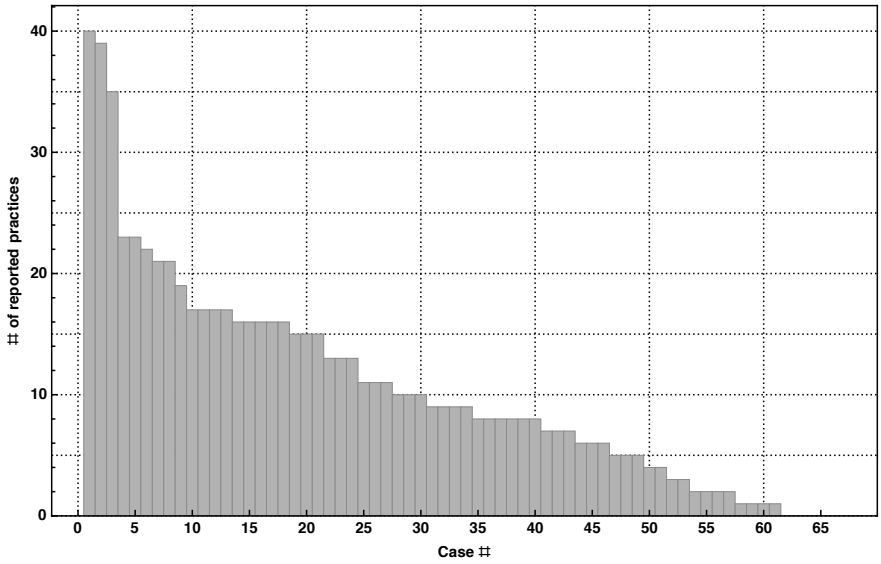

Figure 2. The number of reported agile practices in the studied start-up companies. y-axis show the number of reported practices, x-axis show the studied cases. The cases are sorted by the number of reported practices

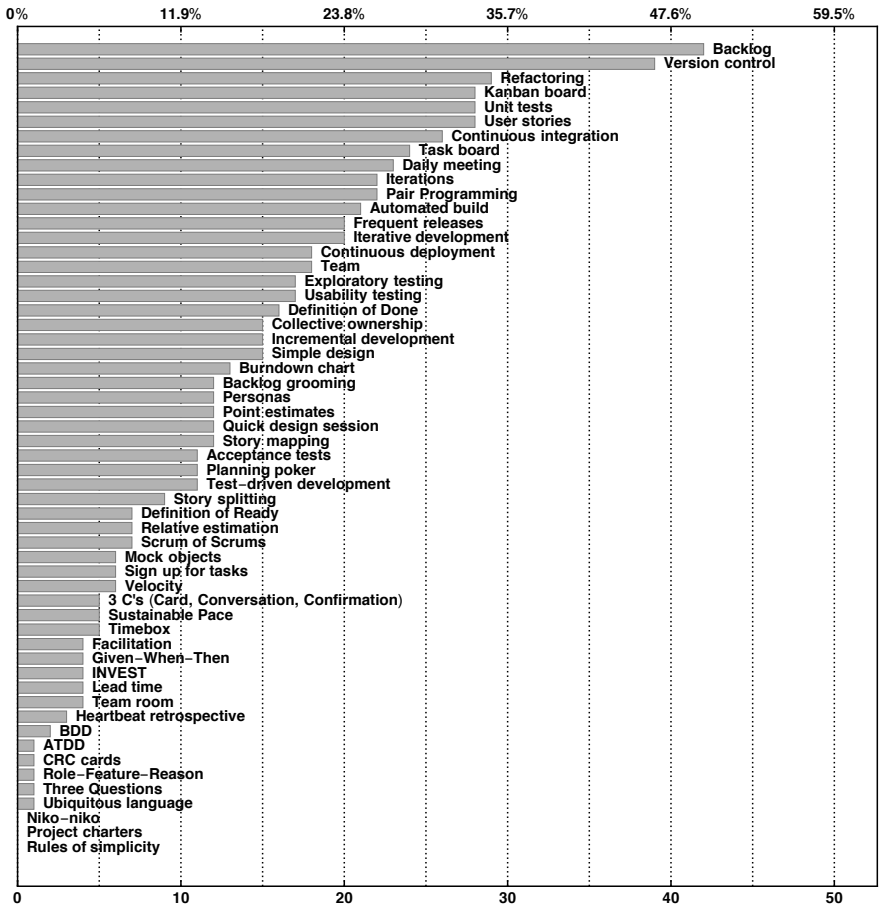

Figure 3. Frequency of Agile practices

Table 2. Results of Cramer's V test on the association between product factors and use of Agile practices with $p < 0.05$. Up (↑) and down (↓) arrows denote whether the association is positive, i.e., use of the practice is associated with more positive responses, or negative, i.e., use of the practice is associated with more negative estimates from respondents

Practice	Documentation	Architecture	Source code	Testing	Overall quality
Card, Conversation, Confirmation	–	–	–	0.422↑	–
Unit tests	–	–	–	0.391↑	–
Automated build	–	–	0.374↑	–	–
Facilitation	–	–	0.330↓	–	–
Given-When-Then	–	–	0.330↓	–	–
INVEST	–	–	0.330↓	–	–
Iterations	–	0.359↑	–	–	–
Continuous integration	–	–	–	–	0.368↑
Collective ownership	–	–	–	–	0.372↓

iterative processes. A few 2 out of 84 follow a waterfall-like process, 10 companies report using an ad hoc approach, see Figure 1.

We presented respondents with a list of 56 Agile practices and asked to tick off the practices that they use in their companies. Most start-ups use between 0 and 20 Agile practices. However, the majority of companies report using only a few practices, see Figure 2. There is also a small cluster of companies reporting the use of more than 35 individual practices. Only 7 companies explicitly reported not using any agile practices, 16 respondents have not provided their answers.

The most frequently used Agile practices are backlog and version control reported by 42 and 39 companies, respectively (50% and 46% out of 84 cases). The use of other practices varies, see Figure 3. Respondents do not report the use of practices such as the Niko-Niko calendar (visualizing the team's mood changes), project charters (a poster with a high level summary of the project), and rules of simplicity (a set of criteria to evaluate source code quality).

4.1. Overview of the findings

In Table 2, we summarize the associations between the use of certain practices and product factors. In Table 3, we summarize the associations between the use of certain practices and team factors. We show only practices with statistically significant associations ($p < 0.05$). The numbers in the table show Cramer's V values denoting the strength of the associations, see Table 1 for interpretation of the values.

4.2. Interpretation of associations

An association shows that a specific practice and certain estimates of a factor are reported together. We use the Pearsons Chi-squared test ($p < 0.05$) to determine if the association is statistically significant. However, from associations alone, we cannot explain the phenomenon with confidence and guide guide practitioners in adopting Agile practices in start-ups. To explain the associations, we formulate 5 archetypes (A) of propositions characterizing the potential explanations of our findings:

It could be that a statistically significant association is a false positive. That is, the association between a practice and a factor is due to an error or some confounding factor.

A_0: There is a spurious association between P and F.

An association could point towards a causal relationship between the use of a practice (P) and

Table 3. Results of Cramer's V test of association ($p < 0.05$) between the use of Agile practices and team factors. Up (↑) and down (↓) arrows denote whether the association is positive, i.e., use of practice is associated with more positive responses, or negative, i.e., use of practice is associated with more negative estimates from respondents

Practice	Attitudes	Pragmatism	Communication	Skills	Resources	Process
Backlog	–	–	–	–		0.401↓
Unit tests	0.379↑	–	–	–	–	–
Continuous integration	0.360↑	–	–	–	–	–
Automated build	–	–	–	–	–	0.346↓
Definition of Done	0.411↓	–	–	–	–	–
Simple design	–	–	–	–	0.365↓	–
Burndown chart	0.383↓	–	–	–	0.384↑	–
Story mapping	–	0.356↑	–	–	–	–
Relative estimation	0.399↓	–	–	–	0.399↑	–
Velocity	0.435↓	–	–	–	–	–
Team room	–	–	–	–	0.343↓	–

a factor (F). We are measuring factors through respondent evaluation, thus we cannot distinguish between actual and perceived improvements.
A_1: Use of P improves perception of F.

Some of the associations appear to be negative, i.e., the use of a practice is reported together with unfavorable estimates. It could be that the practice has adverse effects, or the use of the practice helped to expose the problematic factor:
A_2: Use of P hinders F.
A_3: Use of P exposes issues with F.

It could be that a practice is introduced as a consequence of a situation. That is, we could be observing a reverse causal relationship.
A_4: F is the cause or enabler for using P.

4.3. Specific findings

In this section, we link together our specific findings with relevant propositions, see Figure 4. In the figure we show a list of agile practices with statistically significant associations to factors. The factors are grouped into four blocks A_1–A_4 representing our propositions. The arrows denote potential explanations between factors and practices.

A product backlog is an authoritative list of new features, changes, bug fixes, and other activities that a team may deliver to achieve a specific outcome [24].

Our results show a moderately strong (Cramer's $V = 0.401$) association between the use of a backlog and worse perception of the engineering process. In particular, frequent changes in requirements, unclear objectives, and unsystematic changes hindering the engineering process are reported together with the use of the backlog.

Unit testing is a practice to develop short scripts to automate the examination of low-level behavior of the software [24].

Our findings show a moderately strong association (Cramer's $V = 0.379$) between the use of unit tests and teams' attitudes. In particular, a positive attitudes towards following the best design, coding, and testing practices are reported together with using unit testing.

Our findings also show a moderately strong association (Cramer's $V = 0.391$) between the use of unit testing and less reliance on manual testing of the product.

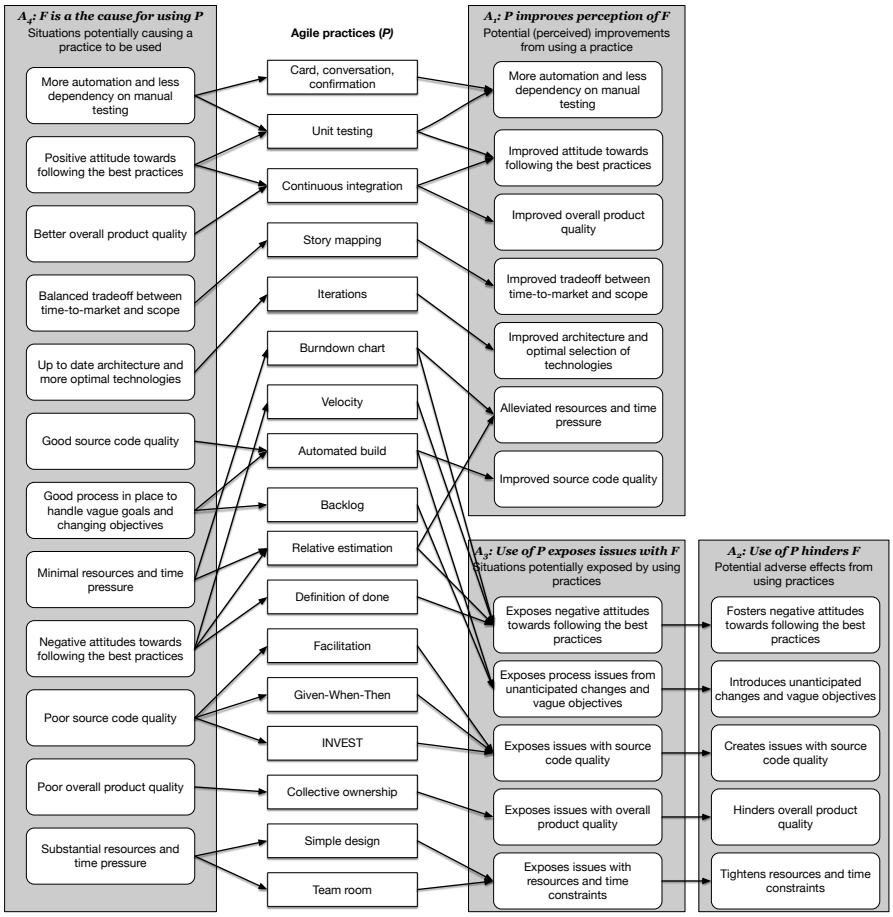

Figure 4. Overview of the findings and the propositions. We show agile practices and different explanations for the associations (A_1–A_4)

Continuous integration aims to minimize the duration and effort of each integration episode and maintain readiness to deliver a complete product at any moment [24].

Our findings show a moderately strong association (Cramer's $V = 0.360$) between the use of continuous integration and more positive attitudes towards using sound design, coding, and testing practices.

Our findings also show a moderately strong association (Cramer's $V = 0.368$) between the use of continuous integration and more positive estimates of product internal and external quality, and less slipped defects.

Automated build is a practice to automate the steps of compiling, linking, and packaging the software for deployment [24].

Our findings show a moderately strong (Cramer's $V = 0.346$) association between the use of automated build and worse estimates in the engineering process.

Our findings also show a moderately strong (Cramer's $V = 0.374$) association between the use of automated builds and more positive estimates on the source code quality.

Definition of done is a list of criteria which a task must meet before it is considered done [24].

Our findings show a moderately strong (Cramer's $V = 0.411$) association between the use of a definition of done and worse attitudes towards following best engineering practices.

Simple design is a practice to favor simple, modular, and reusable software designs that are created as needed [24].

Our findings show a moderately strong association (Cramer's $V = 0.365$) between simple design practices and more pressing time and resource concerns.

Burndown chart is a graph visualizing the remaining work (x-axis) over time (y-axis) [24].

Our findings show a moderately strong association (Cramer's $V = 0.383$) between the use of the burndown chart and worse estimates on teams' attitudes towards following the best engineering practices.

Our findings also show a moderately strong association (Cramer's $V = 0.384$) between the use of the burndown chart and less time and resource pressure.

Story mapping is a practice to organize user stories in a two-dimensional map according to their priority and level of sophistication. Such a map is used to identify requirements for a bare-bones but usable first release, and subsequent levels of increased functionality [24].

Our findings show a moderately strong association (Cramer's $V = 0.356$) between the use of story mapping and a more pragmatic approach to handing the trade-off between time-to-market and following best engineering practices.

Relative estimation comprises of estimating task effort in relation to other similar tasks, and not absolute units [24].

Our findings show a moderately strong association (Cramer's $V = 0.399$) between the use of relative estimation and worse attitudes towards following the best testing, architecture, and coding practices.

Our results also show a moderately strong association (Cramer's $V = 0.399$) between the use of relative estimates and less time and resource pressure.

Velocity is a metric to calculate how long it will take to complete the project based on past performance [24].

Our findings show a moderately strong association (Cramer's $V = 0.435$) between the use of velocity and worse attitudes towards following the best engineering practices.

Team room is a dedicated, secluded, and equipped space for an agile team to collaborate on the project [24].

Our findings show a moderately strong association (Cramer's $V = 0.343$) between theuse of a team room and more pressing time and resource constraints.

Facilitation is a practice to have a dedicated person in the meeting, ensuring effective communication, and maintaining focus on the objectives [24].

Given-When-Then is a template for formulating user stories comprising of some contextual information, triggers or actions, and a set of observable consequences [24].

INVEST is a checklist to evaluate the quality of a user story [24].

Our findings show a moderately strong association (Cramer's $V = 0.330$) between the use of any of the three practices (Facilitation, Given-When-Then, and INVEST) and worse estimates on the product source code quality.

Iterations are time-boxed intervals in an agile project in which the work is organized. The project consists of multiple iterations, tasks, and objectives for the next iteration and is revised just before it starts [24].

Our findings show a moderately strong association (Cramer's $V = 0.359$) between the use of iterations and more positive estimates on the quality of the product architecture. Specifically, respondents report fewer workarounds, more optimal selection of technologies, and fewer issues with outdated designs.

Collective ownership is a practice to empower any developer to modify any part of the project source code [24].

Our findings show a moderately strong association (Cramer's $V = 0.372$) between collective ownership and worse estimates on the product's internal and external quality.

Card, Conversation, Confirmation is a pattern capturing the life cycle of a user story. The life cycle starts with a tangible "card", "conversations" regarding the user story occurs throughout the project; finally, a "confirmation" is received of a successful implementation of the user story [24].

Our findings show a moderately strong association (Cramer's $V = 0.422$) between the use of the life-cycle pattern and less dependence on manual testing of the product.

5. Discussion

5.1. Answers to our research questions

RQ1: How are Agile practices used in start-ups. Our results show that start-ups use Agile practices, even though they do not follow any specific agile methodology. Such results confirm earlier findings, e.g., Giardino et al. [7], and Yau and Murphy [6], stated that engineering practices and processes in start-ups gradually evolve from rudimentary and ad hoc to more systematic.

The most frequently used practices are a backlog, version control, refactoring, user stories, unit tests, and kanban board. We could not identify any clear tendencies comparing the frequencies of practices between different cohorts, e.g., team size, product stage, and team skill levels.

Our results show limited adoption of version control, a backlog, and refactoring. The use of these practices is reported only by 30–50% of cases. Disuse of such practices is reported both by respondents with and without software engineering background. This is surprising, as, e.g., version control is a widely adopted software development practice, can be applied with minimal overhead, and provide substantial benefits [45, 46]. Thus, not benefiting from version control to manage the source code is difficult to excuse with lack of resources, time shifting priorities, or other pressing concerns [3].

The use of Agile practices does not imply that an organization follows agile principles as proposed by the Agile manifesto [14]. Many of the Agile practices, for example, version control, unit testing, and refactoring, among others, could be equally well applied to other types of development methodologies. That said, a majority of start-ups characterize their development methodology as agile. Exploring the maturity of agile processes in start-ups remains a direction for further exploration [9, 47].

RQ2: What are the associations between specific Agile practices and product factors. We identify associations between the use of Agile practices and product architecture, source code quality, test automation, and the overall level of quality. We could not identify any associations regarding the quality and understandability of product documentation.

Practices related to automation, e.g., unit tests, automated build, and continuous integration, are associated with positive estimates of product factors. Practices related to requirements quality, e.g., Given-When-Then, and INVEST, show negative associations. It could be that start-ups introduce such practices as a response to the adverse effects of poor requirements. However, the causal effects of using Agile practices need to be explored further to draw any definitive conclusions.

The use of collective ownership is associated with negative estimates of overall product quality. We propose two interpretations: a) collective ownership exposes the actual state of product internal quality, b) collective ownership has adverse effects.

If two or more developers collaborate on the same part of the product, they may have a more objective view of its flaws. A single developer working on and "owning" a part of a product may be biased in estimating its quality [48].

Alternatively, inviting other developers to work on the part of a product could introduce defects. Other developers, who are not the original authors, may lack the essential contextual information to evaluate and change the component without introducing defects. Practices such as unit testing, continuous integration, and pair programming may help to prevent defects and distribute knowledge in the team. Collective ownership could be an example of a practice that must be supported by other practices to avoid adverse effects.

RQ3: What are the associations between specific Agile practices and team factors. Most associations pertain to teams' attitudes towards the following the best engineering practices. Both positive and negative attitudes towards the best engineering practices are precedents for using several practices. Automation practices, such as unit tests and continuous integration, are associated with positive attitudes. However, control and planning practices, such as the definition of done, burndown chart, relative estimation, and velocity, are associated with negative attitudes to-

wards the following the best engineering practices. We explain such results with the need for tighter control over the team's performance when they do not see the benefits of following best practices.

We observe several associations between the use of Agile practices and respondents' estimates towards time and resource pressures. The use of burndown charts and relative estimates are associated with less pressure. We interpret such findings that the use of resource planning and control practices helps to plan any amount of resources better and alleviate the pressure.

We have not identified any associations about communication in the team. Other authors, e.g., Yau et al. [6] and Sutton [13], have identified that in small start-up teams, communication is not an issue. Small collocated teams do not need additional support for coordination. Such finding leads us to argue that the primary reasons for introducing Agile practices in start-ups are tighter control over a team's performance and resource utilization.

5.2. Implications to research

In this study, we have set forth a number propositions for further investigation. Looking at the propositions summarized in Figure 4, we identify several cross-cutting concerns to address with further studies in the area.

Our results suggest that software start-ups adopt Agile practices one by one without following any particular agile methodology, e.g., scrum or XP. Such finding is supported by earlier work, for example, Giardino et al. [7] and Gralha et al. [49], reporting that new practices are introduced gradually and aimed at addressing specific concerns. However, existing research on adopting agile software engineering considers mostly the adoption of whole methodologies, e.g., scrum or XP, and not individual practices [36]. We identify an opportunity for more fine-grained research on how to adopt Agile practices in small organizations to address their specific concerns.

Our results suggest a limited adoption of ubiquitous engineering practices such as the use of version control, backlog, and refactoring, reported only by 30–50% of cases, see Figure 3. While this result could be explained by the limitations of the survey data collection method, it also suggests a potential disuse of essential software engineering practices in start-ups. This result invites further research to understand how much neglect of the best engineering practices is motivated by the engineering context and how much by the lack of engineering acumen in start-ups.

Related work identifies the need to be more flexible and to alleviate the need for rigorous upfront planning as the primary goal for adopting agile. Other objectives include the aim to improve product quality, shorten feedback loops with customers, and to improve teams' morale [36]. Such objectives are superficial and do not support the adoption of specific practices or addressing specific start-up specific challenges [3]. We identify an opportunity to explore the precedents of introducing specific Agile practices, and longitudinal studies examining the effects of specific practices.

5.3. Implications for practitioners

Examining our findings, we identify several relevant patterns for practitioners.

Teams' attitudes towards following the best engineering practices appear as a strong denominator of adopting a range of Agile practices. Positive attitudes towards good practices drive the adoption of automated testing and continuous integration. Such practices have further positive effects on software quality.

Many respondents, both with and without a software engineering background, failed to report using, for example, version control, refactoring, and unit tests. These are standard software engineering practices, applicable with minimal overhead, and could provide substantial benefits in any software development context.

Negative attitudes towards best practices are associated with the use of practices for progress control, such as the definition of a done, burndown chart, and effort estimation. Our explanation for such a finding is that teams implement such practices to have tighter control over the development process.

Our results suggest that the primary benefits of adopting Agile practices are tighter control

over the team's performance and product quality. The use of progress control practices alleviates resource pressures.

6. Conclusions

In this study, we investigate associations between the use of Agile practices and perceived impact on various product and team factors in software start-ups. Based on our findings, we set forth a number of propositions that narrow down the space of investigation for future studies on Agile practices and start-ups.

We conclude that start-ups adopt Agile practices, however do not follow any specific methodology. The use of Agile practices is associated with improved product quality, more positive attitudes towards following the best engineering practices, and tighter control over resource utilization. However, the exploration of the causal effects remains a direction of further work.

We have formulated several implications for researchers and practitioners. We identify an opportunity for more fine-grained studies (on a practice level) into the adoption and effects of Agile practices. We conclude that Agile practices show a potential to be used in start-ups setting, however, adopting individual practices without considering supporting practices could lead to adverse effects.

7. Acknowledgements

The authors of this paper would like to thank all practitioners who found time and motivation to participate in this study. Reaching this diverse population of start-ups would not be possible without help and support from Software Start-up Research Network[4] community, and specifically Nana Assyne, Anh Nguyen Duc, Ronald Jabangwe, Jorge Melegati, Bajwa Sohaib Shahid, Xiaofeng Wang, Rafael Matone Chanin, and Pekka Abrahamsson.

This work is partially funded by the Foundation for Research Support of the State of Rio Grande do Sul (17/2551-0001/205-4) and the Brazilian National Council for Scientific and Technological Development.

References

[1] C. Giardino, S.S. Bajwa, and X. Wang, "Key challenges in early-stage software startups," in *Agile Processes, in Software Engineering, and Extreme Programming*, Vol. 212, 2015, pp. 52–63.

[2] S. Blank, "Why the lean start up changes everything," *Harvard Business Review*, Vol. 91, No. 5, 2013, p. 64.

[3] C. Giardino, M. Unterkalmsteiner, N. Paternoster, T. Gorschek, and P. Abrahamsson, "What do we know about software development in startups?" *IEEE Software*, Vol. 31, No. 5, 2014, pp. 28–32.

[4] N. Paternoster, C. Giardino, M. Unterkalmsteiner, T. Gorschek, and P. Abrahamsson, "Software development in startup companies: A systematic mapping study," *Information and Software Technology*, Vol. 56, No. 10, 2014, pp. 1200–1218.

[5] E. Klotins, M. Unterkalmsteiner, and T. Gorschek, "Software engineering in start-up companies: An analysis of 88 experience reports," *Empirical Software Engineering*, Vol. 24, No. 1, 2019, pp. 68–102.

[6] A. Yau and C. Murphy, "Is a rigorous agile methodology the best development strategy for small scale tech startups?" University of Pennsylvania Department of Computer and Information Science, Tech. Rep., 2013.

[7] G. Carmine, N. Paternoster, M. Unterkalmsteiner, T. Gorschek, and P. Abrahamsson, "Software development in startup companies: The greenfield startup model," *IEEE Transactions on Software Engineering*, Vol. 42, No. 6, 2016, p. 233.

[8] J. Pantiuchina, M. Mondini, D. Khanna, X. Wang, and P. Abrahamsson, "Are software startups applying agile practices? the state of the practice from a large survey," in *International Conference on Agile Software Development*. Springer, Cham, 2017, pp. 167–183.

[9] E. Klotins, M. Unterkalmsteiner, P. Chatzipetrou, T. Gorschek, R. Prikladniki, N. Tripathi, and L. Pompermaier, "A progression model of software engineering goals, challenges, and practices in start-ups," *IEEE Transactions on Software Engineering*, 2019.

[4]The Software Start-up Research Network, https://softwarestartups.org/

[10] PitchBook Data, Inc., "European middle market report 2h 2015," Tech. Rep., 2015.

[11] PitchBook Data, Inc., "U.S. middle market report Q4 2015," Tech. Rep., 2015.

[12] E. Klotins, "Software start-ups through an empirical lens: are start-ups snowflakes?" in *1st International Workshop on Software-Intensive Business: Start-Ups, Ecosystems and Platforms, SiBW 2018, Espoo, Finland, 3 December 2018*. CEUR-WS, 2018.

[13] S.M. Sutton, E.C. Cubed, and M. Andretti, "The role of process in a software start-up," *IEEE Software*, Vol. 17, No. 4, 2000, pp. 33–39.

[14] K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries et al., "Manifesto for agile software development," 2001.

[15] L. Rising and N.S. Janoff, "The Scrum software development process for small teams," *IEEE Software*, No. August, 2000, pp. 26–32.

[16] V.E. Jyothi and K.N. Rao, "Effective implementation of agile practices-incoordination with lean kanban," *International Journal on Computer Science and Engineering*, Vol. 4, No. 1, 2012, p. 87.

[17] P. Diebold and M. Dahlem, "Agile practices in practice: a mapping study," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. ACM, 2014, p. 30.

[18] S. Jalali and C. Wohlin, "Global software engineering and agile practices: a systematic review," *Journal of software: Evolution and Process*, Vol. 24, No. 6, 2012, pp. 643–659.

[19] T. Chow and D.B. Cao, "A survey study of critical success factors in agile software projects," *Journal of Systems and Software*, Vol. 81, No. 6, 2008, pp. 961–971.

[20] S. Misra, V. Kumar, U. Kumar, K. Fantazy, and M. Akhter, "Agile software development practices: evolution, principles, and criticisms," *International Journal of Quality and Reliability Management*, Vol. 29, No. 9, 2012, pp. 972–980.

[21] A. Solinski and K. Petersen, "Prioritizing agile benefits and limitations in relation to practice usage," *Software Quality Journal*, Vol. 24, No. 2, 2016, pp. 447–482.

[22] E. Mkpojiogu, N. Hashim, A. Al-Sakkaf, and A. Hussain, "Software startups: Motivations for agile adoption," *International Journal of Innovative Technology and Exploring Engineering*, Vol. 8, No. 8S, 2019, pp. 454–459.

[23] R. Souza, L. Rocha, F. Silva, and I. Machado, "Investigating agile practices in software startups," in *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, 2019, pp. 317–321.

[24] Agile Alliance, "Agile glossary," https://www.agilealliance.org/agile101/agile-glossary/, 2018, [Online; accessed 20-April-2018].

[25] L. Layman, L. Williams, and L. Cunningham, "Exploring extreme programming in context: an industrial case study," in *Agile Development Conference*. IEEE, 2004, pp. 32–41.

[26] S. Ilieva, P. Ivanov, and E. Stefanova, "Analyses of an agile methodology implementation," in *Proceedings. 30th Euromicro Conference*. IEEE, 2004, pp. 326–333.

[27] B. Kitchenham and S.L. Pfleeger, "Software quality: the elusive target [special issues section]," *IEEE Software*, Vol. 13, No. 1, 1996, pp. 12–21.

[28] E. Tom, A. Aurum, and R. Vidgen, "An exploration of technical debt," *Journal of Systems and Software*, Vol. 86, No. 6, 2013, pp. 1498–1516.

[29] G. Lucassen, F. Dalpiaz, J.M.E. van der Werf, and S. Brinkkemper, "Forging high-quality user stories: towards a discipline for agile requirements," in *IEEE 23rd international requirements engineering conference (RE)*. IEEE, 2015, pp. 126–135.

[30] R. Khaled, P. Barr, J. Noble, and R. Biddle, "System metaphor in extreme programming: A semiotic approach," in *7th International Workshop on Organizational Semiotics*. Citeseer, 2004.

[31] E.F. Collins and V.F. de Lucena, "Software test automation practices in agile development environment: An industry experience report," in *7th International Workshop on Automation of Software Test (AST)*. IEEE, 2012, pp. 57–63.

[32] R. Moser, P. Abrahamsson, W. Pedrycz, A. Sillitti, and G. Succi, "A case study on the impact of refactoring on quality and productivity in an agile team," in *IFIP Central and East European Conference on Software Engineering Techniques*. Springer, 2007, pp. 252–266.

[33] B. Selic, "Agile documentation, anyone?" *IEEE Software*, Vol. 26, No. 6, 2009.

[34] F. Palomba, G. Bavota, M. Di Penta, R. Oliveto, and A. De Lucia, "Do they really smell bad? A study on developers' perception of bad code smells," in *IEEE International Conference on Software Maintenance and Evolution*. IEEE, 2014, pp. 101–110.

[35] M. Mantyla, J. Vanhanen, and C. Lassenius, "A taxonomy and an initial empirical study of bad smells in code," in *International Conference on Software Maintenance*, 2003, pp. 381–384.

[36] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and Software Technology*, Vol. 50, No. 9-10, 2008, pp. 833–859.

[37] E. Klotins, "Using the case survey method to explore engineering practices in software start-ups," in *Proceedings of the 1st International Workshop on Software Engineering for Startups*. IEEE Press, 2017, pp. 24–26.

[38] R. Larsson, "Case survey methodology: Quantitative analysis of patterns across case studies," *Academy of Management Journal*, Vol. 36, No. 6, 1993, pp. 1515–1546.

[39] A.C. Hope, "A simplified Monte Carlo significance test procedure," *Journal of the Royal Statistical Society. Series B (Methodological)*, 1968, pp. 582–598.

[40] J. Cohen, *Statistical power analysis for the behaviour sciences*. Lawrence Erlbaum Associates, 1988.

[41] A. Agresti, *An introduction to categorical data analysis*. Wiley New York, 1996, Vol. 135.

[42] P. Runeson, M. Höst, A. Rainer, and B. Regnell, *Case study research in software engineering*. John Wiley and Sons, Inc., 2012.

[43] W.W. Daniel, "Spearman rank correlation coefficient," *Applied Nonparametric Statistics*, 1990, pp. 358–365.

[44] M. Crowne, "Why software product startups fail and what to do about it," in *Engineering Management Conference*. Cambridge, UK: IEEE, 2002, pp. 338–343.

[45] B. De Alwis and J. Sillito, "Why are software projects moving from centralized to decentralized version control systems?" in *ICSE Workshop on Cooperative and Human Aspects on Software Engineering*. IEEE, 2009, pp. 36–39.

[46] E. Daka and G. Fraser, "A survey on unit testing practices and problems," in *IEEE 25th International Symposium on Software Reliability Engineering*. IEEE, 2014, pp. 201–211.

[47] L. Gren, R. Torkar, and R. Feldt, "The prospects of a quantitative measurement of agility: A validation study on an agile maturity model," *Journal of Systems and Software*, Vol. 107, 2015, pp. 38–49.

[48] M. Ozer and D. Vogel, "Contextualized relationship between knowledge sharing and performance in software development," *Journal of Management Information Systems*, Vol. 32, No. 2, 2015, pp. 134–161.

[49] C. Gralha, D. Damian, A.I.T. Wasserman, M. Goulão, and J. Araújo, "The evolution of requirements practices in software startups," in *Proceedings of the 40th International Conference on Software Engineering*. ACM, 2018, pp. 823–833.