UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Kristjan Jürisoo

On the Role of Agile Software Development Practices in Software Process Improvement

Master's Thesis (30 ECTS)

Supervisor(s):
Dietmar Pfahl, PhD

Tartu 2017

On the Role of Agile Software Development Practices in Software Process Improvement

Abstract:

Software development in general is still a fairly new and evolving field. Software development companies have been looking for software process improvement (SPI) for years, but the field of SPI is a diverse one. The goal of the Master's thesis at hand is to provide the big picture of the use of agile methodologies along with its practices in the context of SPI within the current literature. Furthermore, the paper at hand aims to provide more insight into the subset of agile practices of a bigger, more general systematic mapping study on the current state of the art of software process improvement. To find answers to the research questions stated in the thesis, systematic literature review was conducted. The results of the literature review were validated by conducting surveys amongst professionals in several Estonian software development companies with varying sizes. The publication rate of papers concerning agile practices in the context of SPI shows a steady rise over the years. Results of the literature review show that the field of agile practices in SPI is fairly evenly balanced regarding being a concept or being adapted and evaluated in the industry. Furthermore, the results indicate that the there are many papers published that are of high relevance to industry professionals. By far the most discussed agile methods in the literature regarding SPI are Scrum and Extreme Programming while the most addressed practices include integrating often, testing first and conducting daily meetings. Within the analysed papers there are plenty of evidence from the industry of using agile methods in SPI. The survey conducted amongst several Estonian companies confirms that the use of agile practices in SPI is a relevant topic and that Scrum being the most popular topic in the literature is not a coincidence.

Keywords:

Agile practices, software process improvement
CERCS: P170

Väledate tarkvaraarendusmeetodite roll tarkvara arendusprotsessi parendamises

Lühikokkuvõte:

Tarkvaraarendus on endiselt võrdlemisi uus ja arenev valdkond. Tarkvaraettevõtted on otsinud tarkvara arendusprotsessi parendamist (ingl *SPI*) aastaid, kuid tegu on mitmekülgse valdkonnaga. Antud magistritöö eesmärk on pakkuda suurt pilti väleda tarkvaraarenduse meetoditest ja tavadest tarkvara arendusprotsessi parendamise kontekstis. Lisaks eelmainitule on antud töö eesmärgiks minna rohkem süvitsi väledate tarkvaraarenduse meetodite alamhulgaga laiahaardelisema kaardistamistöö raames, mis uurib SPI hetkeseisu kirjanduses. Leidmaks vastuseid töös püstitatud uuritavatele küsimustele, viidi antud töö raames läbi süstemaatiline kirjanduse ülevaade. Kirjanduse ülevaate tulemused valideeriti erinevates Eesti tarkvaraettevõtetes viies läbi küsitlusi. Avaldatud teoste, mis hõlmavad tarkvaraarenduse meetodite kasutamist SPI kontekstis aastane avaldamishulk on vaikselt kasvanud aastate jooksul. Kirjandusülevaate tulemused viitavad, et väledate meetodite kasutamine SPI jaoks pole ainult idee. Avaldatud teoste hulgas on võrdlemisi võrdselt teoseid, mis analüüsivad antud teemat idee tasandil ning neid, mis kirjeldavad väledate tarkvaraarenduse meetodite kasutamist ka professionaalses töös. Lisaks sellele viitavad tulemused, et professionaalide jaoks potentsiaalselt kasulikke teoseid on kirjanduses suur kogus. Enimuuritud väleda tarkvaraarenduse meetodid SPI kontekstis on *Scrum* ning *Extreme Programming*. Kõige enam mainitakse kirjanduses tavasid nagu *pidev integratsioon*, *kõigepealt testi* ning *igapäevaste koosolekute pidamine*. Kogu uuritud kirjanduses on piisvalt tõendeid tööstusest väledate meetodite kasutamisest SPI tarbeks, mida kinnistavad ka Eestis opereerivates tarkvaraettevõtetes läbiviidud küsitluse tulemused.

Võtmesõnad:

Väleda tarkvaraarenduse meetodid, tarkvara arendusprotsessi parendamine
CERCS: P170

Table of Contents

1 Introduction

Software development is an industry that is about 50 years old. This means it is still a young industry and constantly evolving. Software development companies have been looking for software process improvement (SPI) now already for decades [1]. Systematic mapping study done in 2015 [2] suggests that SPI is a diverse field. The study addresses topics like SPI success factors, new trends in SPI and SPI employing agility in improvement process. There are several SPI standards like CMMI and ISO/IEC 15504, but complying to these standards oftentimes requires a lot resources, which most of the companies do not have in abundance. To tackle these issues, the use of agile approaches in general has grown over the years. How and when to use agile approaches has been a question since the introduction of the approach.

1.1 Problem Statement

SPI is a wide research subject that has been researched for some time. Although having been researched for a long time, not much is known how agile practices relate to SPI. The thesis at hand aims to address the use of agile methodologies with its practices in software process improvement. The objective of this thesis is to provide a bigger picture of how agile is seen in the context of SPI - which methods and practices are more used in the context of SPI and how much evidence of using agile practices in SPI is reported from the industry.

1.2 Context

This thesis is part of a more comprehensive SPI related mapping study [2] that aims to address the whole field of SPI and is referenced as *main study* in this research. The results of that study show the constant publication of papers related to SPI. Even though papers are published constantly they lack specific models and theories. The mapping study [2] was clustered into more specific topics, of which one concerns agile methodologies in SPI. The thesis at hand uses this subset of the study as a starting point to analyse the topic of agile methods and practices in SPI in a more detailed way by conducting a systematic review of all the published papers in the subset of the main study. The subset is made up of 73 papers that have been classified as being related to agility in SPI. The papers are received as .zip file containing pdf-s of the papers as well as an Excel spreadsheet with all the titles, authors and years of the papers. The validity of the classification of the 73 papers is revisited and reanalysed in this thesis. The final number of papers involved might change during the analysis process.

The main study [2] was conducted as a follow-up to a previous study [3] on the same topic. For the main study in this paper at hand, the authors took lessons learned from the initial study and used it to develop an update strategy to be able to keep track of the SPI field over time. The enhanced strategy changed the procedure of data collection and improved paper classification strategy. The main study comprises of 769 papers which were carefully picked after completing a search based on 11 complex search strings, applying inclusion and exclusion criteria, voting process and integration with the previous study's dataset. For instance, the inclusion criteria that included the selected paper's title, keyword list and abstract made it explicit that the paper was related to SPI or the paper presents SPI related topics, like models, assessments, etc. Excluded were studies that were not in English or not related to computer science in general, papers that occurred more than once or were not available for download. The analysis and classification of papers was only done on the final dataset of 769 papers.

1.3 Research questions

1.3.1 RQ 1. What is the study population on SPI with special focus on agile?
This research question aims to answer the question of when are the publications of papers made, are they relevant in the industry and what sort of papers they are – solution proposal, opinion paper, evaluation research, etc.

1.3.2 RQ 2. Which agile methods, practices and techniques are addressed in SPI?
This research question aims to answer the question of which agile methodologies, practices and techniques are used in the context of SPI based on a predefined list of agile practices and elements.

1.3.3 RQ 3. To what extent has the use of agile methods in the context of SPI been validated?
This research question aims to answer the question of how well evaluated and tested are agile techniques in SPI and to what extent.

1.3.4 RQ 4. Is the literature relevant in the industry?
This research question is designed to find out if the results of the survey are relevant in the industry – whether the addressed topics in the literature are talked about topics amongst industry professionals too and can the literature be of use for industry professionals.

1.4 Research design

The study at hand is divided into two parts. The first part involves conducting a systematic literature review and the second research part involves validating the results of the first part in the industry by interviewing professionals in selected software companies. The topics of the industry survey are influenced by the literature survey in order to validate the findings. In other words, the outcomes of the first part of this work contribute to the input of the second part. The results of the second part give an indication if the literature is in line with the current state in the industry. The methodologies used for conducting the literature survey and interviews amongst professionals are more explained in the next section of this work.

1.5 Methodology

The current thesis is following a systematic review instrument [4] to provide more insight into bigger, more general mapping study [2] that this study is a subset to. The main study was conducted as a mapping study as opposed to following a systematic review instrument in this study. The main differences between the two lie in the fact that mapping studies have a much broader scope – the research questions are more general, the results for search terms involve much more papers, the analysis is likely to involve overall statistics and distribution of papers rather than in-depth analysis techniques and the dissemination is more aimed at pointing the directions for future studies [4].

The paper at hand aims to deliver more specific insights into the topic while confirming or denying the initial classification of publications. The main study was conducted following the guidelines proposed by Petersen et al. [5], but since the thesis at hand is aimed to be more specific and insightful, it follows systematic review instrument that is described by Kitchenham and Charters [4]. The selection of this particular review instrument is due to the fact that other subsets of the main study also take advantage of guidelines presented in this review instrument which provides comparability between the studies of different subsets.

1.5.1 Systematic review instrument

"A systematic literature review (often referred to as a systematic review) is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest." [4] The main idea behind doing a systematic literature review (SLR) is to take advantage of already existing knowledge and not duplicating the same kind of things. SLR can be used for different reasons:

- For encapsulating existing knowledge concerning a certain technology or method
- For finding knowledge about a certain field to locate missing points of views for conducting more extensive research in the same field
- For providing background for a research problem
- For providing empirical evidence or showing the lack of it to a hypothesis

The present study takes advantage of some of the proposals for conducting a SLR presented in a study that was published in 2007 by Kitchenham and Charters. They set out to create a collection of guidelines based on existing guidelines to provide means for a fair evaluation of research in software engineering field. The proposed systematic review instrument originates from medical research review instruments, meetings with domain experts and different textbooks describing systematic review principles. [4]

In the frame of this study the methods for evaluating and assessing primary studies (individual papers that contribute to the systematic review) several proposed steps are used: defining the research questions (see section 1.3), defining the data to be extracted from each primary study and inclusion/exclusion criteria as well as maintaining the list of included and excluded studies. Following the process of evaluating each individual study, the trends and usage of agile practices in the frame of SPI are analysed.

1.5.2 Paper inclusion and exclusion

In the main study the classification and meta data attribute assignation was mainly performed by using the abstract. This leaves a possibility that the current subset used in this study includes papers that are not relevant to the objectives of this study – not addressing agile practices in the frame of SPI. This is the reason why inclusion and exclusion criteria are used in this study to filter the initial data set (see Table 1). Papers not available in the initial given set of files are additionally searched for from public academic libraries by the title and authors that are retrieved from the spreadsheet that is included in the initial given .zip file. These libraries include:

- ACM Digital Library[1]
- SpringerLink[2]
- ScienceDirect[3]
- Google Scholar[4]

Criteria	Description
EC1	Paper is not available

[1] http://dl.acm.org/

[2] https://link.springer.com/

[3] http://www.sciencedirect.com/

[4] https://scholar.google.com/

EC2	Paper is a duplicate
EC3	Paper includes no relation to agile methods
EC4	Paper is not in the field of SPI in general
EC5	Paper is not in English
IC1	Paper is related to SPI and includes references to agile methods and practices or agile thinking in general

Table 1. Exclusion and inclusion criteria of papers

1.5.3 Rigor and Relevance

Every paper in this study will have a quantifiable scoring applied to it concerning its rigor and relevance, as proposed by Ivarsson and Gorschek [6]. It is done to be able to summarise the state-of-research and measure its relevance to the industry. Ivarsson and Gorschek argue that one of the traditional methods of the relevance or impact of a research - the number of citations - does not necessarily mirror the success in the industry. Based on the available literature they came up with a different method to evaluating rigor and industrial relevance of papers in software engineering. The underlying goal of evaluating papers by rigor and relevance in this context is to provide an overview of the state-of-research, rather than in-depth analysis. For example, scoring how context of the paper is reported can be done via different aspects – tools used, people, market, product, etc. Detailed classification of aspects can lead to few papers scoring high, which, in turn, limits the evaluation of the state of research field in the big picture. Ivarsson and Gorschek propose to evaluate papers by aspects that are often reported in the current field of research. [6] The descriptions of the scoring aspects are described next.

It has been argued that for a research to be relevant and transferable to industry it should be close to reality [7]. To increase the realism of a research it should consider in the evaluation the aspects of scale, context and subjects used. Scale considers the size and the time scale of the application, context is a reference to the environment of the evaluation and the subjects used refer to the people carrying out the evaluation. All of these aspects can have substantial results on the transferability of the evaluation into the industry. Using small instances over a short span of time for a technology under the evaluation might not give enough insight of how to scale an application to industry level of usage [8]. The context of the technology under evaluation should be described, otherwise practitioners are not able to evaluate the suitability of the evaluation [9]. Moreover, the subjects used in the evaluation can also have impact on the relevance of the evaluation for the industry since they might not behave in the way professionals would [10].

The model proposed by Ivarsson and Gorschek [6] aims to provide means to classify evaluations in order to characterize state of research. The proposed model is applicable to many types of evaluations. The model consists of rating papers per its relevance as being relevant (value 1) or not being relevant (0) in 4 different aspects: subjects, context, scale and research method.

Papers can also be evaluated and quantified by rigor which can refer to the precision of the research method used or the correct use of any method. In the context of the study at hand, rigor refers to the extent to which the aspects concerning rigor are presented in the paper rather than the actual rigor of the papers. If the study is not presented adequately the rigor of the paper becomes irrelevant even if it was conducted in a rigorous way. For easy replicability, the study context along with study design needs to be described, because artificial environment is likely to influence the motivation and goals of the research. Scoring papers in terms of rigor follows similar methods as relevance. Three aspects (context described, study design described, validity discussed) need to be evaluated as strong (value 1), medium (0.5) or weak (0). Three aspect levels make it possible to uncover papers that mention aspects related to rigor but don't describe them sufficiently. [6]

Different types of studies have different study designs, different context. For example, a paper proposing a framework, evaluating a tool in practice or giving guidelines will all probably have different study setup. Moreover, when scoring relevance, the subjects should be intended users of the technology according the original model. When researchers are proposing a framework, and evaluating it on professionals, the subjects will be counted as relevant. To be able to roughly quantify all the papers based on the same scoring system, some clarifications to the original rigor and relevance scoring system are outlined in Table 1 and 2. The outlining contributes to the replicability of the process by others and making it possible to better understand the scoring process in general. The original detailed scoring guidelines proposed by Ivarsson and Gorschek are reported in Appendix I.

Attribute	Score of 1	Score of 0,5	Score of 0
Context	The development context (motivations, reasons, development mode, background) is understandably and thoroughly described and can be compared to another context.	The development context is mentioned and can be compared to another context.	There is no context description and the context is not comparable to others.
Study design	Study design is described to an extent that it is replicable and the reader can understand the setup of the study including its measured variables, sampling size, controls used, etc.	The description of study design is vague, making the study not easily replicable.	No description of the study design.
Validity	Are different validity threats discussed thoroughly?	Threats are mentioned, but not described thoroughly.	No discussion of threats to validity.

Table 2. Scoring of rigor

Attribute	Score of 1	Score of 0
Subjects	The subjects of the study evaluation are professionals, the intended users of the technology or the technology is proposed by a professional.	The subjects are researchers, students or not mentioned at all.
Context	The evaluation or proposal is conducted in industrial setting.	The evaluation or proposal is conducted in laboratory or other artificial setting.

Scale	The evaluation is done in a realistic size, contributing the industry.	The evaluation is done partially or at a scale that is not relevant for the industry.
Research method	The method used in the research considers real situations, e.g. case study, field study, exploratory study, action research, etc.	The method used in the research does not consider real situations, e.g. conceptual analysis, laboratory experiment, etc.

Table 3. Scoring of relevance

To sum it up, rigor describes how is an evaluation performed and reported. Whereas relevance shows the potential impact of a research to the industry and academia. It should be noted that relevance refers to the potential impact not the actual impact. [6]

1.5.4 Research type facet

Wieringa et al. [11] propose a model to classify papers and a set of evaluation criteria for evaluating each class. The purpose of paper classification is meant to provide basis for evaluating papers of different nature and content. Not all papers can be evaluated based on the same criteria.
Wieringa et al. [11] developed this model based on what they call *engineering cycle* which consists of 5 different non-sequential engineering tasks – problem investigation, solution design, solution validation, solution selection, solution implementation and implementation evaluation. Based on these activities in the engineering cycle they then propose classifications that help classify papers as those that describe research activates, a design activity or other relevant activities related to the engineering cycle. In the main study Kuhrmann et al. suggest to use the following research type facets:

- Evaluation research
  Evaluation research studies investigate a problem in practice or an implementation of a technique in practice. The evaluation requires more than just one demonstrating case study. Evaluation research papers should be evaluated based on the claimable knowledge and the soundness of the research method rather than the novelty of the technique.
- Solution proposal
  Proposal of solution type of papers propose a technique as a solution for a problem along with argumentation for its relevance. The proposed technique should be a significant improvement on an existing one or a completely new, novel technique. The technique should be presented along with proof-of-concept that could be a sound argument, a small example, or other convincing mean. The evaluation of these kinds of papers should take into account whether the technique is clearly explained, is it novel and sound, has it got wider relevance and if the competitors are discussed.
- Philosophical papers
  Philosophical papers propose a new perspective of things or a new conceptual framework, new way of thinking. They should be evaluated by their originality, soundness and insightfulness.
- Opinion papers

These papers include opinions of the author of how something should be done or what is good or bad about something. Since these kinds of papers involve opinion the evaluation should consider the amount of discussion it provokes and how sound or surprising is the stated opinion.

- Experience papers
  As the name suggests, these papers emphasise *what*, not *why*. It is important that the experience is the author's personal and that the paper included lessons learned along with reports of usage. For evaluating these kinds of papers one should analyse the originality of the experience and soundness and relevance of the report.

The aforementioned categories are a slightly modified version of the original categories that were designed by authors for requirements engineering (RE) papers. For the paper at hand the facets proposed by Kuhrmann et al. are used for the classification process to be in-line with the main study.

In the current paper, these research type facets are used to provide a general overview of the dataset used in the study. It should be noted that one paper can belong to more than one facet, although some combinations are unlikely. Assignation of research type facet provides means to have an overview of what type of research has been conducted in the field which, in turn, can help make conclusions about the state of agile practices in the frame of SPI.

1.5.5 Contribution type facet

Back in 2003 Shaw [12] did a research about the number of research papers submitted to ICSE[5] 2002 to find out about the acceptance rate of papers by the committee and how to better design research projects and write papers. One of the outcomes was a list of different types of papers. For best comparability with the main study and other sub studies of the main study similar types of contribution type facets are used as in the main study [2]. The full list can be seen in Table 4.

Type	**Description**
Model	Generalisation and conceptualisation of concepts
Theory	Construct of cause-effect relationships
Framework	Framework or method related to agile methods in SPI
Guideline	List of advices
Lessons learned	Set of outcomes from results
Advice	Recommendation
Tool	Agile tool that embodies a technique

Table 4. Contribution type facets [2]

In the scope of the study at hand, applying contribution type facets contributes to understanding the state-of-research. Along with research type facets the classification helps de-

[5] http://www.icse-conferences.org/

termine what type of research has been conducted in the research field. For instance, if a framework-classified paper is a solution proposal type of paper or philosophical paper. The difference being the former including an evaluation of the proposal as well.

1.5.6 New metadata attributes

In the main study, all the papers have metadata attributes assigned to them. The study at hand is namely the set of all papers having a metadata attribute *agile*. To have a closer look in terms of agile methods, new metadata attributes are created to better map the current situation in the literature. The metadata attributes are selected based on literature and include the most popular and used agile methods along with its practices and elements. All the attributes are tabulated using Microsoft Excel, which helps create a visual overview of the content of the papers via heat map. Slethot et al. [13] proposed 35 agile practices of which 12 originate from the Scrum methodology and 23 from Extreme Programming (XP). Some of these practices are overlapping and used in both methodologies. The practices were picked while conducting a literature review in a systematic manner which provides a good starting point for the current study.

The 35 agile practices of XP and Scrum are adjusted to better fit the data and augmented with additional practices that originate from other agile methodologies. This helps cover the whole field of research and also give an overview of the position in literature about the less used agile practices. Moreover, practices or elements that are identified in the papers while analysing them, are added to the metadata attribute heat map.

On top of analysing specific practices of different methods, the usage of methods in general is also analysed. On top of Scrum and XP, other methods mentioned in the papers are also mapped. Furthermore, Lean software development approach and Kanban method are included into this list because of their close relations to the agile world.

The definitions and descriptions of the practices and methods in general is presented in sections 2.1. and 3.2 of the paper at hand.

All in all, the overall picture of the analysis procedure is depicted on Figure 1. Procedure of systematic literature review at hand. The initial dataset includes a list of 73 papers from the main study with metadata attribute *agile*. The first round of analysis comprises of applying the inclusion and exclusion criteria outlined in section 1.5.2 of this paper to the papers received as an input to this study. This also helps to identify missing papers and outline the need for searching them in public libraries. After cleaning the initial dataset, more thorough and extensive analysis is done according to the review protocol described in previous paragraphs. All the papers are analysed for their rigor, relevance, research and contribution type facets, publication year, agile methods and practices mentioned, country performed in and organisation size. The aforementioned data is inputted into a spreadsheet to have an overview. From the spreadsheet, all the agile methods and practices will be mapped to a heat map that is predefined as outlined in section 1.5.6 of the paper at hand. Finally, state-of-research and results are derived from the visualisation of data.

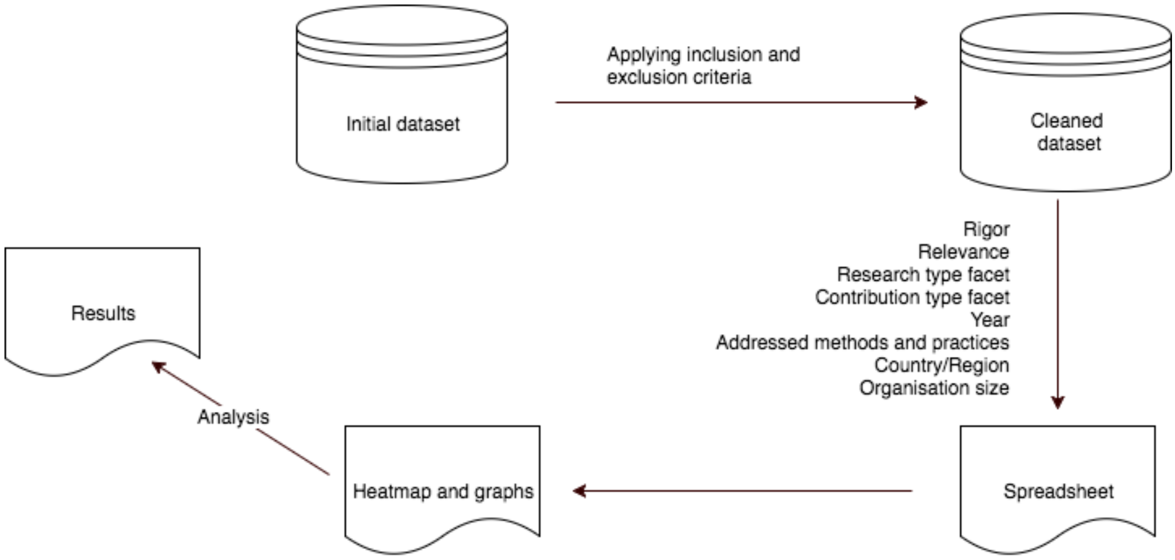


Figure 1. Procedure of systematic literature review at hand

1.5.7 Industry survey

The second part of the work in hand – the industry survey – is meant to validate the findings of the literature survey rather than be a thorough case study on the topic. The survey has quantitative and qualitative questions and, in general, is designed as a short questionnaire to get as many different responses as possible. Questions with quantitative answers provide good basis for conducting a visual and quantifiable analysis by the researcher. On top of quantitative questions, qualitative questions are used to gather more insight from the respondents of the survey since qualitative questions give the respondent freedom of input. The survey questions are based on the literature survey outcomes. To find out if the most addressed topics in the literature are relevant topics in the industry, there are questions finding this out. Likewise, for the least addressed practices. More specifically about conducting the survey in section 3.4.1 of this work.

2 Background

This section of the work gives an insight into agile methodologies and software process improvement – two integral parts of the study at hand. It is necessary to have a clear background and description of all the definitions used in this papers. Especially since agile methods evolve and change overtime. First part of this paragraph describes agile methods and the second part focuses on software process improvement (SPI).

2.1 Agile methods

The official starting point for agile methods was formally written down in 2001. The document is called Agile Manifesto and was written by 17 software practitioners [14]. They outline 4 main values and 12 principles. The four main values are:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

The need for agile approach came into prominence in mid-1990s when it became clear that there was need for an alternative approach for software development. It became evident that software development needed to be more responsive to change. The roots of agile approaches started appearing in the late 1980s when it became obvious that a more flexible approach to software developing was need because of the perceived long-windedness of the used structured methods [15]. The pressure of getting things done more quickly led to the adoption of *rapid application development* (RAD) approaches. RAD approaches were meant to use spiral system development lifecycle which reduces the time it takes to fix errors, but at the same time makes the system more prone to errors.

Agile approaches divide the system to be developed into iterations, meaning that functionality is added with each iteration. This approach makes it possible to not have full specification of the system at the start of the project, which is especially relevant for customers that explore new business areas and don't know the end result in detail. One of the main differences of agile approaches compared to conventional ones, like Waterfall, is that rework and iterations are encouraged and not to be avoided. [15]

There are different schools of thoughts concerning the development of agile approaches. One suggests that agile methods should only be used in a stable environment whereas other suggest that they can be successfully used in environments where requirements and scope are not well-defined at first. But what is agreed upon, is that the success of using agile approaches depends on the people practicing it and their ability to make decisions without explicit approval from their senior managers. Moreover, essential parts of agile approaches include the business viability of the project, constant testing and reversibility of changes. Agile practices are meant to be lightweight, orientated to involve customer at every point of incremental development and adaptable to possible changes in requirements. The readiness to respond to change along with continuously delivering business value are the focus points of agile methods. [15] [16]

The first agile methods like Adaptive Software Development (ASD), Crystal, Dynamic Systems Development method (DSDM), Extreme Programming (XP), Scrum and Feature Driven Development (FDD) were developed in late 1990s [18]. The main contributors behind these methods were also responsible for the publication of Agile Manifesto. All current agile methods share common ideas and emphasize the four main values of agile methods.

The mapping of 35 agile practices described in section 1.5.6 for analysis of the papers include practices from Scrum and Extreme Programming methods. The following paragraphs describe more in detail these methods. Furthermore, Lean software development and Kanban are briefly introduced since these methods are also mapped during the analysis process as described in 1.5.6.

2.1.1 Scrum

Scrum originates from the USA, but the roots of the name lie in rugby that comes from Britain. In software development Scrum refers to an agile framework for managing the development of a product. The essence of Scrum is its incremental and iterative approach to development process. [15] Scrum is an agile project management method and does not say how one should produce a product. Scrum is by far the most used agile methodology around. According to a survey [17] Scrum is used by 56% of software developing companies.

Scrum defines *sprints* during which a functionality of a product is developed, tested and implemented. At the beginning of each sprint the team selects a collection of tasks that can be turned into shippable functionality at the *spring planning meeting*. Sprints include 15-minute *daily meetings* with team members and can last up to 30 days. The daily meetings are meant for teams to collaborate, keep track of every member's progress and, if necessary, make changes in the process or tasks. The selection of tasks during sprints is handled by pulling *user stories* from *product backlog*, the features list. Another important part of Scrum is that all elements of Scrum project are time-boxed, which means they have certain timeframes.

The Scrum framework defines three roles - *Product Owner*, *Scrum Master* and the development team. Role of the project manager, called Scrum Master, is different in Scrum compared to more conventional teams since teams are self-managing and project managing as such involves different tasks. Role of the Scrum Master is to make sure all participants adhere to the Scrum process. It means that Scrum Master is owner of the process rather than the product. Every Scrum project team also has a product owner who maintains the product backlog and its prioritises to deliver strong value to the customer.

At the end of every sprint the whole team holds a *review meeting* where the functionality of the product is presented to the product owner and other stakeholders. One of the intentions of this meeting is also to get people together and plan for the next move. After the review meeting another meeting – *Sprint retrospective* – is held. Sprint retrospective is organised by the Scrum Master and encourages teams to revise the development process and make it more effective.

2.1.2 Extreme Programming (XP)

Extreme programming or XP as it is often called, is, as the name suggests, an extreme approach to agile approaches. Extreme programming concentrates more on the implementation of software rather than project management like Scrum. Moreover, XP emphasises team work as all members of the team are equal. Idea of Extreme Programming is to tackle relatively small projects with tight time schedule in small, tightly focused development teams. The development process in XP teams is somewhat different to other agile approaches since it involves two developers working at the same computer, called *pair programming*, while reviewing each other's code and releasing frequently. XP also values testing and test driven development (TDD) as an important part of the method. The implementation is created, tested and implemented in tight contact with the customer who must be available all the time. [15]

2.1.3 Lean software development

Lean software development (LeanSD) shares its values with agile software development methods. In fact, lean and agile methods have been found to be similar. LeanSD is an approach where traditional Lean manufacturing practices of mainly reducing waste are applied to software development. The principles used in Lean software development are [19]:

- Eliminate waste
- Amplify learning
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
- Build integrity in
- See the whole

2.1.4 Kanban

Kanban is a lean approach to agile software development. Similarly, to Scrum, Kanban encourages breaking work into small deliverables and self-organising teams, for example. But most of all Kanban is a visualisation practice – write tasks on a paper and put them on the wall. By creating and naming columns the wall, one can see the workflow and identify causes of delays in the development process visually. Every column, or workflow state, is limited to a capacity, called Limit Work in Progress (WIP). Moving tasks into a state is done by pulling rather than pushing and can only be done while there is capacity in the state. [19, 20]

2.2 Software Process Improvement (SPI)

Software Process Improvement (SPI) is an initiative to improve the processes that define the development of software. Software process is the set of actions along with tools, methods and practices that are used in the development of software. To improve the software processes, the organisation conducting it needs to understand the current status of the development processes, develop a vision of the required process, establish a list of required process improvement actions, produce a plan to accomplish it, commit resources and start all over. [21]

In order to organise and continuously improve their development processes and the capabilities of teams, companies look for SPI. There are several different SPI models that companies can take advantage of: CMMI[6] (Capability Maturity Model Integration) and ISO/IEC 15504[7], also known as SPICE (Software Process Improvement and Capability determination). CMMI consists of 17 core process areas of which each is made up of goals and practices how to reach the goals. SPICE includes 5 process categories which all consist of 9 process attributes that are assessed on a four-point rating scale. Compliance to the models is indicated by the maturity level in the case of CMMI or capability level in the case of SPICE. In addition to these reference models, there are goal-driven approaches that can be used for driving SPI, such as GQM (Goal-Question-Metric) [23] and deployment paradigms like QIP (Quality Improvement paradigm) [22] and IDEAL (Initiating, Diagnosing, Establishing, Acting, Learning) [24]. SPI models are focused on organisational

[6] http://cmmiinstitute.com/

[7] https://www.iso.org/standard/60555.html

level and outline *what to do* whereas agile methods concentrate on team level and outline *how to do* [25].
The aforementioned SPI models were first released at end the last century, so they were designed for top-down development approaches that were popular at the time. The introduction of agile methods and their fundamental differences to traditional methods brought along the need for SPI models to adapt to agile methods. Traditional software development process in SPI sees a universal, repeatable solution to provide predictability whereas agile approach is flexible and aims to provide faster development times, increased customer satisfaction and responsiveness to change. Moreover, the traditional approach relies on document based knowledge transfer, but agile approach on face-to-face communication. Similar discrepancies are highlighted by the differences of immediate focus of process improvement – agile development in SPI aims to improve current daily working practices, on the other hand, traditional development aims to improve organisational processes in future projects. Initiatives for mapping and combining of SPI models and agile methods have been done for some time now and it has shown that CMMI-DEV and agile can work together, since they work on different abstraction levels. [25, 26]

3 Analysis and Findings

This section provides an overview of the findings and results.

3.1 RQ 1. Study population

The following sections give an overview of the analysed papers and their general attributes.

3.1.1 Exclusion and inclusion of papers

The first task of analysing the input data received from the main study concerns validating the papers as being papers that address SPI in the context of agile methods or practices. To achieve that, exclusion and inclusion criteria described in section 1.5.2 are applied. In total, 10 papers were missing from the initial given folder with paper files. The 10 missing papers were searched for from public libraries. After querying public databases, 8 more papers were found, out of which one was excluded because it was written in Spanish. Out of the 73 papers 14 papers are not relevant for the current study or are not available in the initial dataset (see Table 5). The final dataset to be analysed comprises of 59 papers.

Reason	Count
EC1	2
EC2	1
EC3	8
EC4	2
EC5	1

Table 5. Reasons for paper exclusion from further analysis

3.1.2 Publication frequency

The study set includes 59 papers. The first paper to discuss agility in SPI was released already in 1996. Figure 2 gives an overview of the publication frequency and rates over 20 years up to 2015. There was a small peak of papers published 3 years after releasing the Agile Manifesto, but by far the biggest number of papers were released in years 2013 and 2014, 13 and 10 respectively (see Figure 1). Before the release of the Agile Manifesto there was only 1 paper published regarding agile thinking in the frame of software process improvement.

The paper published before Agile Manifesto, released by M. Aoyama in 1996 [27], describes concurrent developing process in Japan which includes practices like *dividing software into smaller parts*, *time-fixed interval of delivery*, *close customer relations* and *incremental construction* of the system. These practices are very similar to the agile practices described nowadays, like *time-boxed sprints producing shippable output* in Scrum and *on-site customer* in XP and Scrum. That is also the reason why these papers are included in the analysis even though they were published before the release of agile methodologies as we know today.

In general, the average publication rate has grown over the years, although publications have come in 2 or 3 year waves. 2004 and 2008 stand out with 4 publications compared to years before and after those years. From 2010 to 2014 the publication rate has been higher, on average almost 8 papers per year. Years 2012 and 2015 seem to confirm the wavy nature of publications, with 3 and 2 publications respectively.

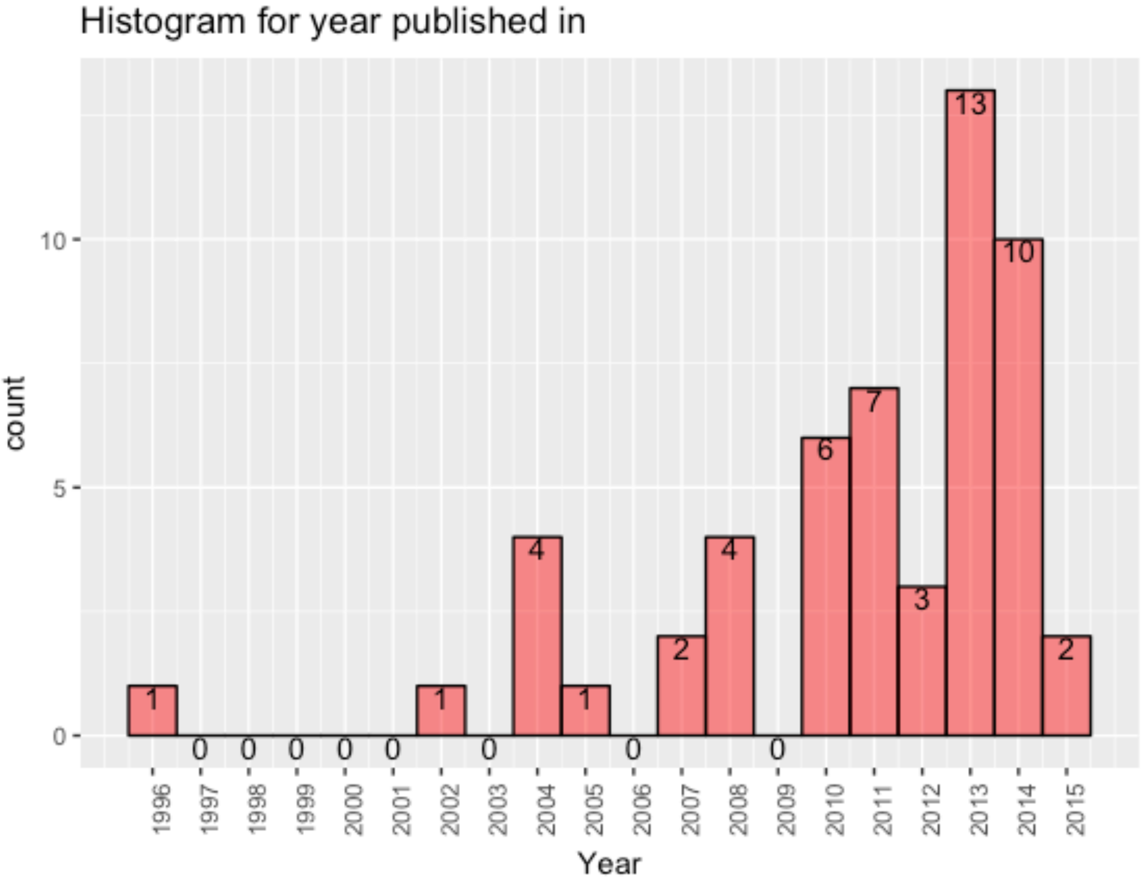


Figure 2. Number of publications over years

3.1.3 Research and Contribution Type facets

Figure 3 depicts the contribution [2] and research type facets [11] of analysed papers. Out of the 59 papers, 30 papers (51%) report lessons learned, out of which 12 papers are experience papers, 13 papers are philosophical papers and 6 evaluation researches. 43% of lessons learned papers being philosophical suggests that these lessons are learned from artificial setting or concluded from secondary studies. In the total dataset of 59 papers, 12 are experience papers out of which 11 are lessons learned with the odd one being an advice paper. Not a single paper is an opinion paper. The distribution by contribution type amongst philosophical papers is somewhat more distributed than amongst experience papers. The 13 philosophical lessons learned papers make up 54% of all philosophical articles. Tools and guidelines both contribute 3 philosophical papers whereas 5 frameworks are proposed in a philosophical paper, meaning that these frameworks are not evaluated. Out of all 18 framework proposals (these include method proposals) 6 are evaluation papers and 7 are solution proposal, suggesting that 33% of these papers are evaluated in industry, but 39% are solution proposal still waiting to be evaluated. Only 3 model papers are amongst the analysed data. 2 of these are solution proposals and only one of them is evaluated.

In summary, the contribution and research type facets indicate that agility in SPI is quite evenly balanced regarding being a concept or being adapted and evaluated in the industry. Philosophical papers and solution proposal make up 56% of all papers and evaluation researches with experience papers make up 44%. On the other hand, the data is clearly distorted towards lessons learned (51%) and frameworks (31%) which make up about 80% of all papers, meaning that tools and models are not a relevant study area in the research field of agility in SPI. Same can be said about providing guidelines and advice.

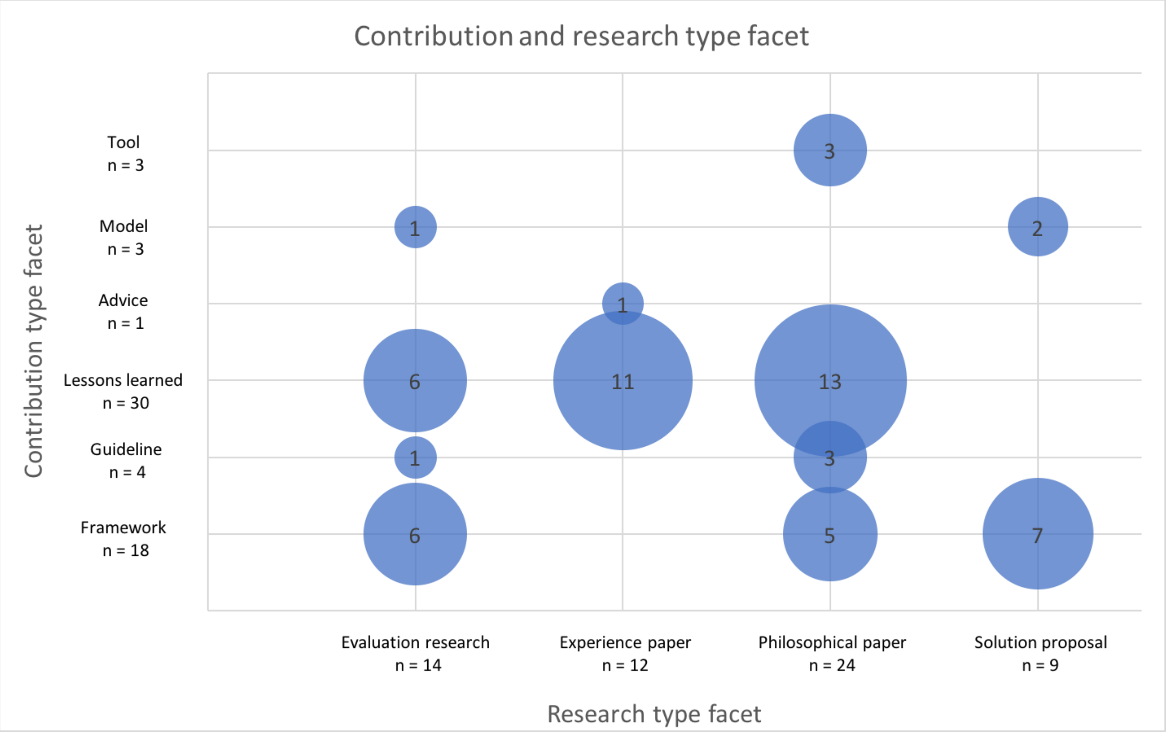


Figure 3. Contribution and research type facets of analysed papers

3.1.4 Rigor and Relevance

The overview of applying rigor and relevance [6] to each paper can be seen on Figure 4. It clearly indicates that the biggest single group of papers is the group with the highest rigor in applied research methods and highest relevance to the industry. These papers make up about 17% of all papers (10 out of 59). In fact, a total of 46% of analysed papers (27 in total) are of the highest relevance to the industry, suggesting that all of these papers are reporting knowledge that is applicable to practice. These papers usually are case study papers that report an experience or evaluate, for instance, a framework or method that practitioners can take note of, and apply to their own practice. At the other end of the spectrum are papers that have little or no relevance to professionals whatsoever. Papers with relevance of 1 or 0 make up a total of 18 papers (31%). These papers usually include philosophical papers that provide a new perspective of things, but don't evaluate their ideas, making them not directly applicable for industry professionals.

In general, papers with higher relevance tend to score higher also in terms of rigor. This could be because of the fact that papers that do an evaluation in industry or in a setting that it is appropriate for the proposal or experience, also describe the context, study design and validity more than papers that are, say, philosophical papers describing a new view on a certain subject.

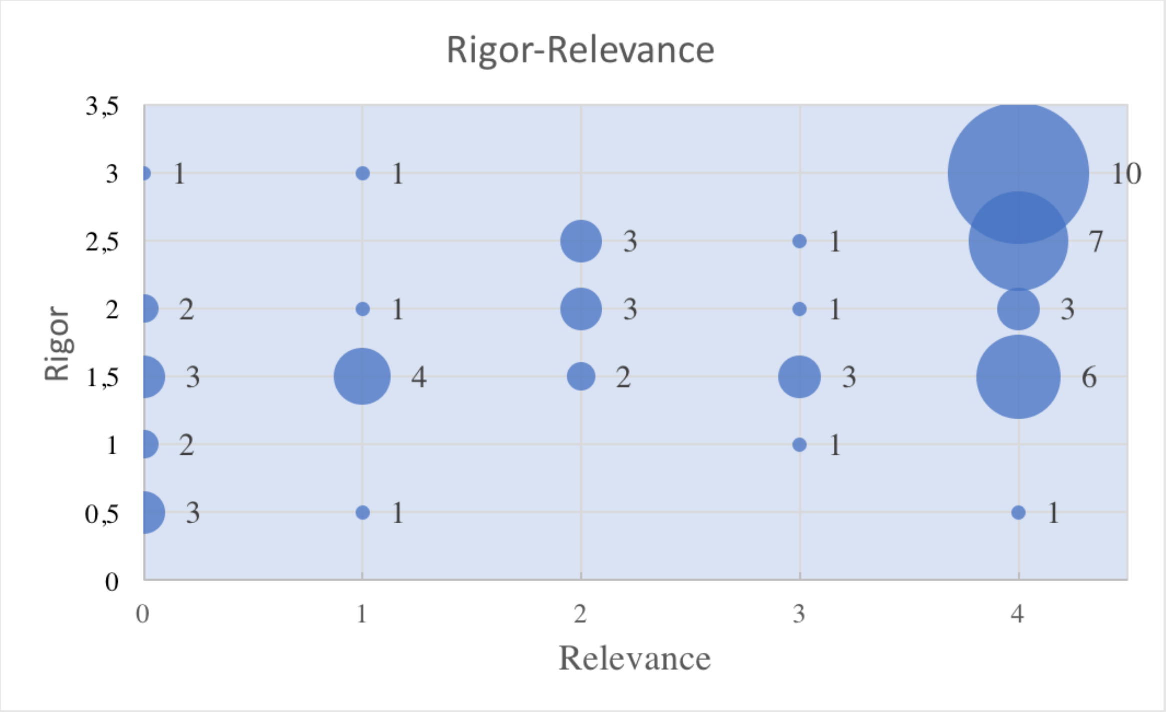


Figure 4. Relevance and research method rigor of the analysed papers

3.2 RQ 2. Agile methods and practices in SPI

The following paragraphs give an overview of the addressed methods and practices in the analysed literature.

3.2.1 Agile methods

Out of the 59 analysed papers 5 do not explicitly mention any agile methods in regards to SPI. The remaining 54 papers, however, do address agile methods by their name. Figure 5 shows the distribution of methods addressed. Scrum and XP are by far the most researched methods. More than three quarter of papers (46 out of 59) address Scrum in the context of SPI and about 71% (41 out of 59) XP. The overwhelming dominance of Scrum and XP is in line with the prevalent nature of these methods in practice. Following Scrum and XP with the count of papers being mentioned in, is the Lean approach to software development. Lean approach in the frame of SPI is mentioned in 14 publications out of 59, being little under fourth of all publications. Other methods like Crystal, Feature Driven Development and Dynamic Systems Development method are all addressed in around 13-18% of papers, with 8, 9 and 11 mentions respectively. The least mentioned of all agile approaches is the Adaptive Software Development method with just 5 papers addressing this approach. Similarly to ASD, Kanban visualisation techniques are discussed as a topic in 6 papers.

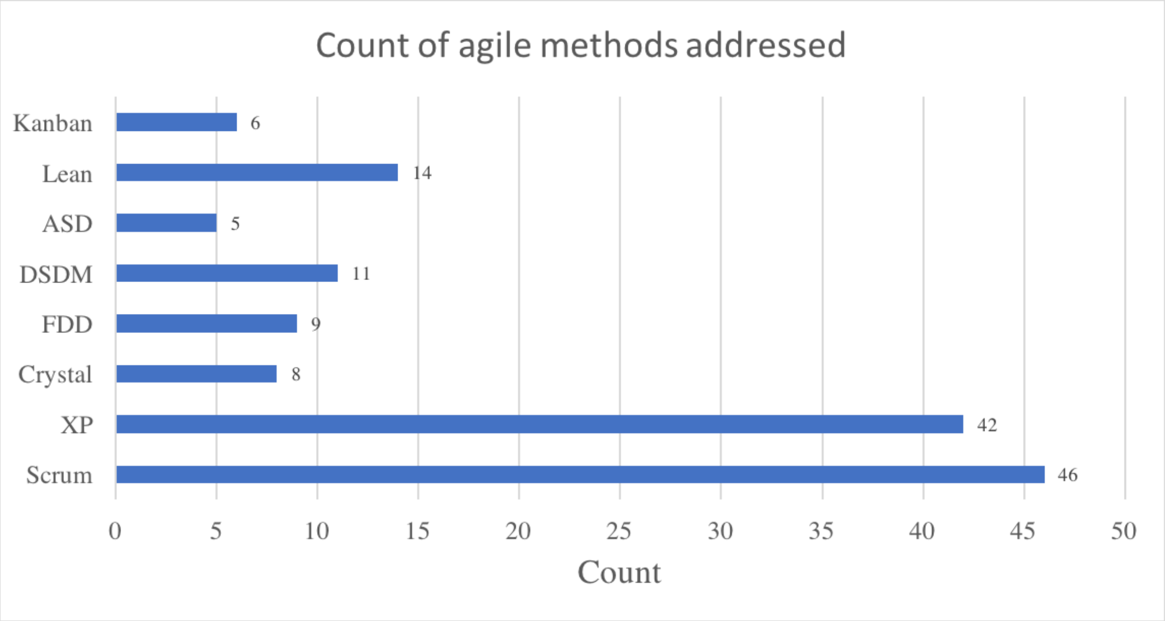


Figure 5. Agile methods addressed in analysed papers concerning SPI

In order to understand if any method has been started to be talked about more over the years, the mapping of each paper's publication year over each agile method is utilised. The resulting heat map can be seen on Figure 6. At first sight the map can be misleading since the years with most mentions of methods are also the years with most publications (see Figure 2). 2013 and 2014 had 13 and 10 publications respectively. In both of these years only one paper does not concern Scrum and three papers do not address XP. Extending this analysis to other years, similar results can be seen. In fact, only year 2004 was a year where Scrum was not addressed by more than half of the publications. In 2004, one paper out of four addressed Scrum, whereas all papers in that particular year talked about XP. Throughout all years, XP, similarly to Scrum, remains constantly talked about. Looking at other methods, the publication rate suggests that neither of those has become significantly more or less researched. Crystal, DSDM and Lean were all addressed in 1 paper out of 4 in 2004 whereas the count in the years with most publications stayed between 1 and 5, suggesting that these methods are still researched, but at the same time have not grown in popularity, with mentions still being in less than 50% of papers. In this data Crystal and ASD are the only methods that don't have a substantial rise of publication rate in the more populated years of 2013 and 2014.

		1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
Method	Scrum										1			1	4		5	5	2	17	9	2
	XP								1		4	1			3		4	5	2	13	7	2
	Crystal										1			1	1			1		1	2	1
	FDD														1			1	1	2	4	1
	DSDM										1				1			1		3	4	1
	ASD														1			1		1	1	
	Lean		1								1						1	1		4	5	1
	Kanban																			2	4	

Figure 6. Agile methods addressed in analysed papers concerning SPI by paper publication year

The small selection number makes it impossible to have far-fetching generalisations based on this data, but it can be observed that none of the methods has fallen out of research topics and none of the methods has gained sudden interest.

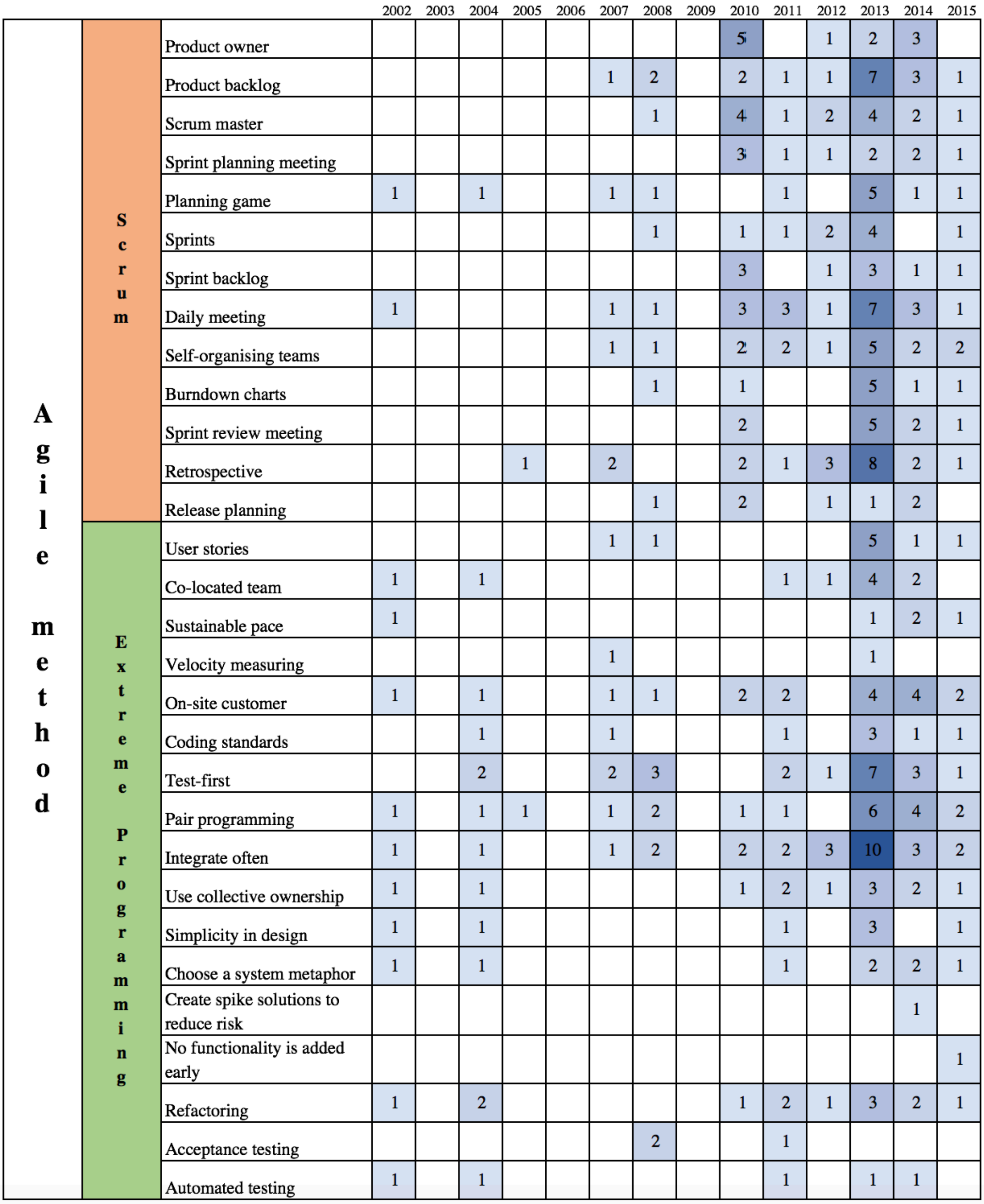
3.2.2 Agile practices

The dominance of Scrum and XP in the literature also contributes to the selection of which agile practices to map for analysing in the context of SPI. Figure 7 gives an overview of addressed agile practices in the input data of this study. To give a better visual overview, practices with no mentions in the papers were removed from the map, as were the years prior to the first mention of a practice. These included practices like *Move people around*, *When a bug is found tests are created*, etc. For further better visualisation and general mapping of topics addressed, the initial list taken as the starting point [13] was somewhat reduced in its wording to be more general. For example, practice *All production code is pair programmed* was changed to *Pair programming*, *Team members volunteer for tasks (Self-organizing team)* to *Self-organising teams*, etc. Furthermore, some terms include practices that have the same meaning but different wording in the papers. For instance, the practice *Give the team a dedicated open work space* was made more general by calling it *Co-located team* and includes terms used in the papers like *sit together, open workspace* and *team collaboration*. Finally, 30 practices were included in the mapping.

Figure 7 and Figure 8 show *daily meetings*, *testing first*, *pair programming*, *retrospectives* and *integrating often* in the frame of SPI as researched topics. These practices are also continuously researched over time and the focus has not intensified nor hindered.

Looking more closely at one of the practices, pair programming, it can be observed that in [28] [29] [30] [31] [32] [33] it is discussed how pair programming can contribute to software process improvement initiative, [33] gives a brief mention of how pair programming can contribute to SPI but do not analyse it more empirically and [34] [35] address pair programming in the frame of agile adoption.

Two agile practices are only addressed once. These include *creating spike solutions to reduce risks* and *adding no functionality early*. Other practice that receives a very brief mention is *velocity measuring* with two mentions. While proposing a reference model for agile quality assurance in [36] the adherence to agile practices and CMMI process area of the model's process areas is shown. In [36] the authors propose a reference model for agile quality assurance. While doing so, they also map the relation of their model's process areas, CMMI's process areas and agile practices which include creating spike solutions. For conducting SPI within individual agile project teams, the authors of [26] propose an Iterative Improvement Process. They mention velocity measuring as a practice used to provide basis for time estimation of tasks in the following iteration. While proposing a framework for describing the agile software development maturing process [37], the authors ratiocinate from related work that adding no functionality early is proved to be a difficult practice to assess, and therefore should be assessed by conversation and observation. However, this is not evaluated empirically in their study.



			2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
Agile method	Scrum	Product owner									5		1	2	3	
		Product backlog						1	2		2	1	1	7	3	1
		Scrum master							1		4	1	2	4	2	1
		Sprint planning meeting									3	1	1	2	2	1
		Planning game	1		1			1	1			1		5	1	1
		Sprints							1		1	1	2	4		1
		Sprint backlog									3		1	3	1	1
		Daily meeting	1					1	1		3	3	1	7	3	1
		Self-organising teams						1	1		2	2	1	5	2	2
		Burndown charts							1		1			5	1	1
		Sprint review meeting									2			5	2	1
		Retrospective				1		2			2	1	3	8	2	1
		Release planning							1		2		1	1	2	
	Extreme Programming	User stories						1	1					5	1	1
		Co-located team	1		1							1	1	4	2	
		Sustainable pace	1											1	2	1
		Velocity measuring						1						1		
		On-site customer	1		1			1	1		2	2		4	4	2
		Coding standards			1			1				1		3	1	1
		Test-first			2			2	3			2	1	7	3	1
		Pair programming	1		1	1		1	2		1	1		6	4	2
		Integrate often	1		1			1	2		2	2	3	10	3	2
		Use collective ownership	1		1						1	2	1	3	2	1
		Simplicity in design	1		1							1		3		1
		Choose a system metaphor	1		1							1		2	2	1
		Create spike solutions to reduce risk													1	
		No functionality is added early														1
		Refactoring	1		2						1	2	1	3	2	1
		Acceptance testing							2			1				
		Automated testing	1		1							1		1	1	

Figure 7. Agile practices addressed in the frame of SPI over time

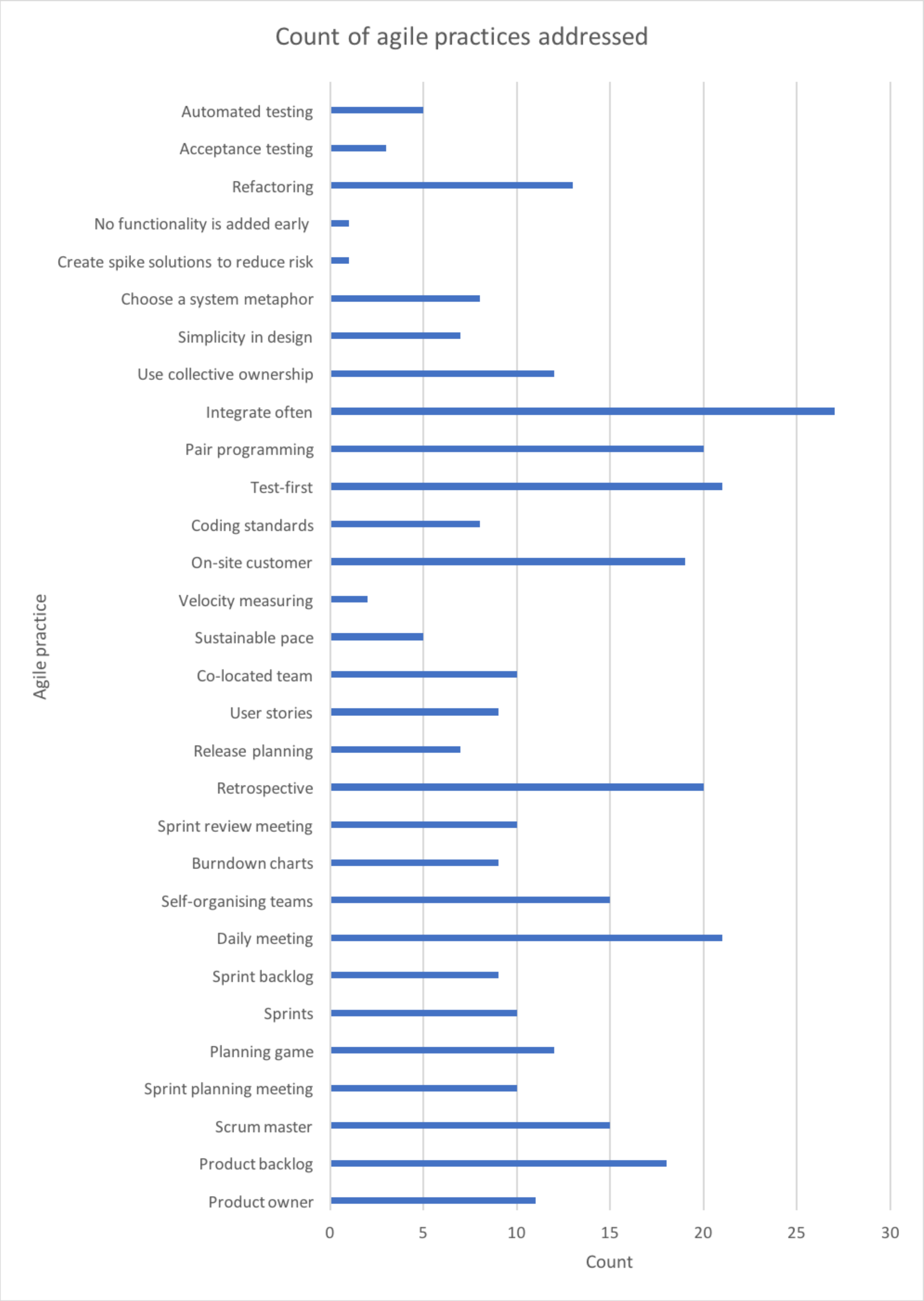


Figure 8. Total count of agile practices addressed in the frame of SPI

3.3 RQ 3. Extent of evidence of using agile methods in SPI

The main focus of finding evidence of using agile methods and practices in SPI is at evaluation researches, experience papers, papers that include case studies or some other sort of professional assessment and papers with high relevance score since these papers should have an impact on the industry.

For instance, [38] is a solution proposal paper proposing lightweight assessment methods for automotive SMEs. The assessment of their proposal is done in an Irish company with 8 development staff. Their assessment method recommended a mixture of plan-driven and agile development practices for SPI. The agile practices that the company adopted included *burn-down charts*, *stand-up meetings*, *continuous integration* as well as *incremental software development*. [39] outlines experiences of taking advantage of agile principles and practices like *unit testing* and *test first* when improving processes at Daimler-Chrysler. Both aforementioned papers outline how agile practices were used in the frame of SPI, but also say that further results of the success are to be published in the future. The Irish automotive company plans to spread the use of agile to their parent company, whereas Daimler-Chrysler is spreading agile practices to other units of the company.

[40] reports transition from plan-driven process to Scrum and the effects of it on software quality. The authors compare 17-month-long plan-driven project and 20-month agile project within a large Norwegian company and conclude that Scrum made managing software defects better and improve the success rate of the project, but at the same time makes developers feel more stress and does not lead to lower defect density. They say that *short sprints* help resolve issues quickly and that *daily Scrum meetings* help share knowledge, which in turn improves defect fixing efficiency.

SPI initiative at R&D unit of Ericsson in Italy made use of agile method fragment repository to guide their agile adoption [29]. They used the repository for getting information about different agile practices like *on-site customer*, *sprint planning*, *short releases* and *daily scrum meeting*. However, from the paper it remains unclear to what extent was the information used or how successful was the SPI initiative.

Two XP projects in a Finnish research centre VTT Electronics [30] showcase regular project level process improvements after every iteration with post-iteration workshops. Although not conducted in full industrial setting, they suggest that negative and positive findings decreased with every workshop and that the immediate visibility of SPI actions increased satisfaction of the team. Positive findings outlined in the workshops included *pair programming*, *continuous integration*, *on-site customer*, *refactoring* and *planning game* whereas negative findings included practices like *code commenting*, *effort estimation*, *TDD* and *testing*. The same research centre also published a longitudinal case study regarding the value of SPI knowledge gained from project level and used at organisational level [31]. They outline *retrospectives* as the point where project level experience is passed on to organisational level for conducting SPI.

The authors of [41] used 5 small Egyptian companies to validate process increment approach to SPI. A process increment is an improvement chunk which can be adopted in a relatively small amount of time. They report that the validation process in the companies worked well – the companies had sustained improvement velocity, teams had high morale and senior management had better project visibility. Moreover, they found out that most of the improvements were implemented by developers, which disconfirms some managers' position that developers are not interested in SPI or too busy for it. The results in these companies were obtained by using agile techniques such as *process cards, iterative lifecycle*, *product backlog*, *information radiator* and *burn-up charts*.

[42] presents a tool to improve company's processes and implementing a set of agile project management practices by using CMMI-DEV 1.2 as a reference model without mentioning any of the practices explicitly. Although the validation in three small companies in Mexico suggests that the tool can be used to strengthen defined agile practices to achieve high CMMI-DEV capability levels, the authors don't use agile practices as the driver of SPI initiative. The authors conclude that the qualitative study needs augmentation from quantitative study.

To support the evaluation, adoption and improvement of agile methods, the authors of [32] proposed a framework. Their case studies on two different pilot projects in the industry suggest that step-by-step adoption of agile processes leads to positive results and that agile practices are appropriate for large and complex projects. The first case study outlines the positive results of using *release planning* and *iteration planning* along with *test-first* approach. The second case study started out with *pair programming* from XP to familiarise the company with agile approach, but soon realised that this is not suitable for them. Pair programming was enhanced and modified, which resulted in self-organising teams that applied *enhanced pair programming*, *pair review* along with *on-site developer*. The use of these practices resulted in reduced production cost, reduced duration and improved product quality to name a few. This study perfectly sums up agile approach to software development – practices are tailored to best suit the project or organisation for generating business value.

One of the most wide-ranging studies in the dataset of the study at hand comes from Nokia [43]. More than 1000 respondents from all over the world participated in survey mapping the adoption of agile practices. Although the adoption of agile practices was seen as a big challenge by many respondents, 60% of respondents saw improving their processes by going over to agile practices as a good and irreversible process and only 9% wanted to return to old development processes.

Most of the aforementioned studies report industry experiences from pilot projects or smaller units inside a company. This could be a risk for drawing conclusions from these studies. However, all of the reported experiences from industry plan on using more agile practices in their SPI programs because of the positive effects. This clearly indicates that using agile practices for guiding SPI initiatives in a company, whether a SME or big corporation, has positive outcomes and is more and more relevant in the software development industry.

3.4 RQ 4. Validation of findings in industry

In order to validate the findings of the study at hand, a survey was conducted amongst several Estonian software development companies.

3.4.1 Conducting the survey

The questionnaire contains several questions of qualitative and quantitative nature (see Appendix II for specific questions). The main goal of the survey is to validate the findings of the systematic literature review – whether the results are in line with the industry or not. The results of section 3 of the current study suggest that practices like *integrate often (continuous integration)*, *daily meetings*, *retrospectives*, *pair programming*, *test first* are the most talked about and addressed practices in the context of SPI. The survey should give an answer if this is the case in the industry as well. The same goes for the least addressed popular agile practices such as *adding no functionality early, creating spike solutions, velocity measuring*.

To have a somewhat generalizable set of answers, several different software development companies of varying sizes were picked as potential respondents to the survey. For best results in terms of the quality of responses, the position of the targeted respondents within the companies should be that of Head of IT/Engineering or other with very good knowledge of the software processes at the company.

Since the survey is not designed to be a thorough case study, but rather a brief survey for validating the results of the current study, the survey is designed to be answered in a short time. To have a better study results validation process, the survey is designed in a way that the response rate amongst the potential respondents would be as high as possible. First couple of questions of the survey are meant for gathering background information about the companies and the second part of the survey aims to tackle more specific topics. To approach the chosen companies, personal acquaintances of the author of the current study are asked for a relevant respondent within the company to maximise the response rate.

Approached companies operating in Estonia:

- TransferWise
- LHV
- CGI
- Mooncascade
- Pipedrive
- Net Group
- Veriff
- Taxify
- Codeborne
- Voicecom

The survey is conducted using Google Forms[8]. It has two questions with qualitative answer fields, three with multiple choice, one with select-boxes and one question with 19 statements to be assessed using five-point Likert scale.

3.4.2 Results of the survey

During the survey process the respondents had the opportunity to keep their company anonymous, so all the reported results are also anonymous. Out of the 10 approached companies, answers were received from 7 companies. The answers were gathered over the course of a two-week period in the summer of 2017. Figure 9 depicts the sizes of the companies that responded to the survey. The spectre of the company sizes should provide a good basis for the survey. Responses were received from a company having as many as 700 people and as well from a company of just 7 employees. 5 companies have between 20 and 85 people and one company around 300 people working for them.

First substantive question of the survey was meant to uncover the knowledge of the respondents regarding SPI initiatives in general. The question asked the practitioners what they considered a SPI initiative to be. Only couple of meaningful responses were received, which stated that an SPI initiative is something during which software processes are improved. Others decided not to answer to this question. One of the reasons could be that the wording of the question was not understandable enough for all the respondents. Other explanation could be that the respondents have never came across an explicitly defined software process improvement initiative as is the case when assessing CMMI levels for example.

[8] https://www.google.com/forms/about/

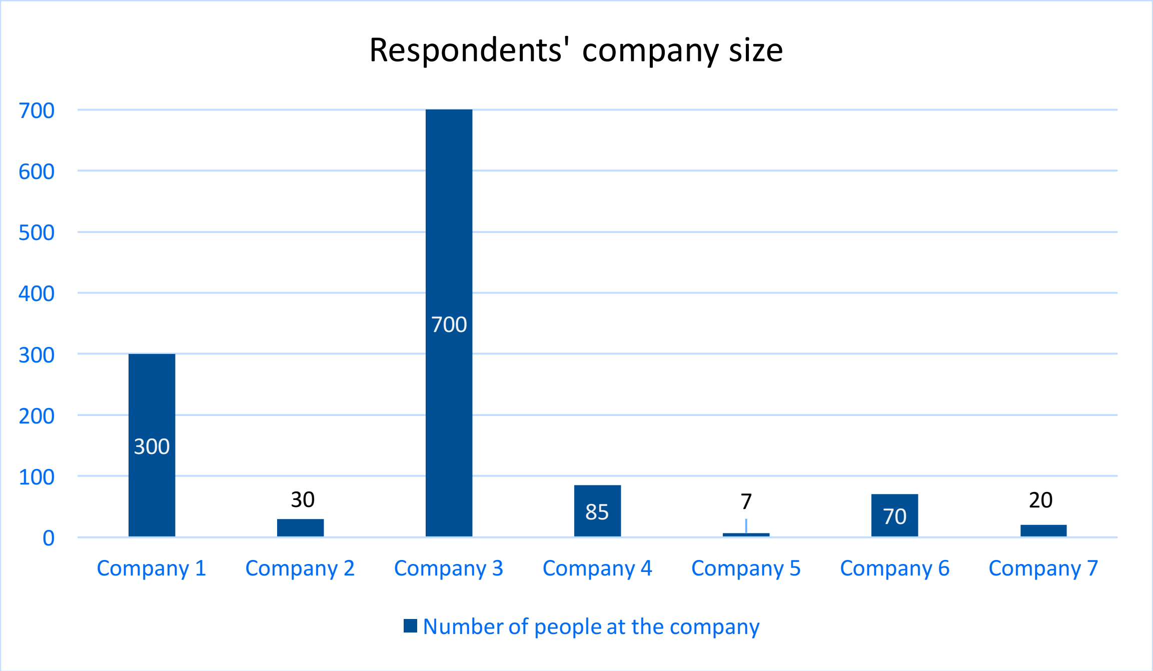


Figure 9. Number of people working at the companies of survey respondents

Third question of the survey asked for the agile methods used in the corresponding companies to see if the methods used in the companies are in line with the findings regarding the methods addressed in the literature (see Figure 5). Out of the 7 companies 5 reported using elements of Scrum. Scrum was also the most talked about agile method within the analysed literature. In the literature, Extreme Programming is almost as addressed topic as Scrum, but this does not seem to be the case amongst the companies surveyed. Two out of seven companies mention using practices from XP, whereas five companies report using Kanban practices. Lean Software Development is used by one company and other methods get no mentions at all (see also Figure 10).

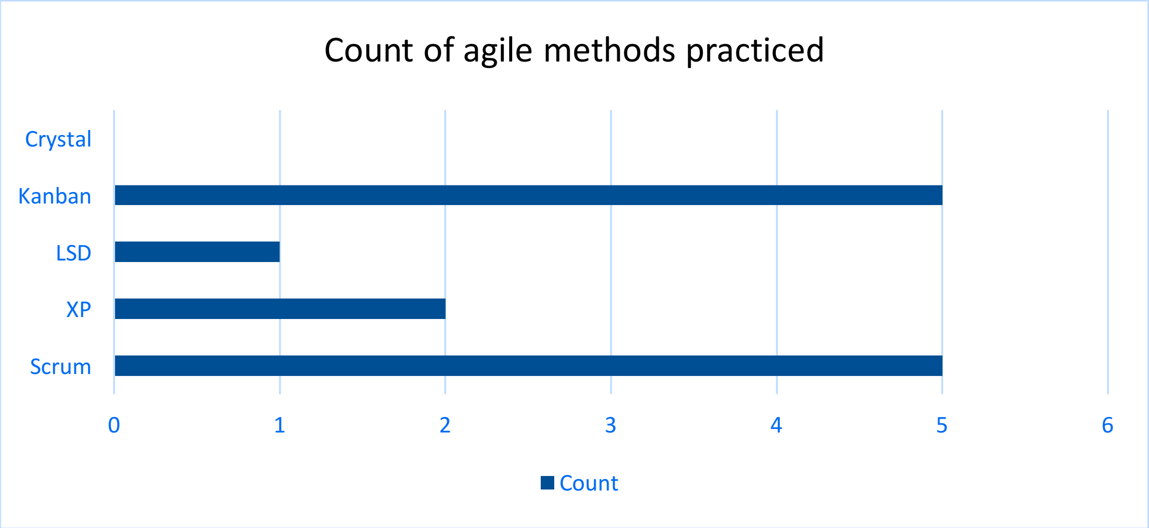


Figure 10. Count of agile methods used in the surveyed companies

Interestingly, the use of Kanban techniques in Estonian organisations surveyed in this thesis is not in-line with the literature. From the literature, Kanban seems to be a not so researched topic, especially in the frame of SPI, but it certainly has an important place in the processes of the surveyed companies.

The use of agile practices within the companies that were surveyed in this study is depicted on Figure 11. 5 out of the 7 companies report using *retrospectives* in their software processes. As can be seen from Figure 8, the use of retrospectives in the context of SPI is one of the most addressed topics in the literature as well. Similar pattern can be seen with the practice of *daily meetings*, with 6 companies reporting the use of daily meetings in their current software processes as well as *test first* approach which is used by 5 companies.
Integrating often and *pair programming* are also one of the most talked about practices in the analysed literature. Within the companies surveyed in this study, 3 out of 7 and 4 out of 7, respectively, reported using these practices. The fact that these practices are used in about half of the companies surveyed, means that these topics are relevant in the industry. The use of a particular practice in a company is also down to the nature of the company and its products. For example, it might not be feasible to use pair programming in a company with just 7 employees.

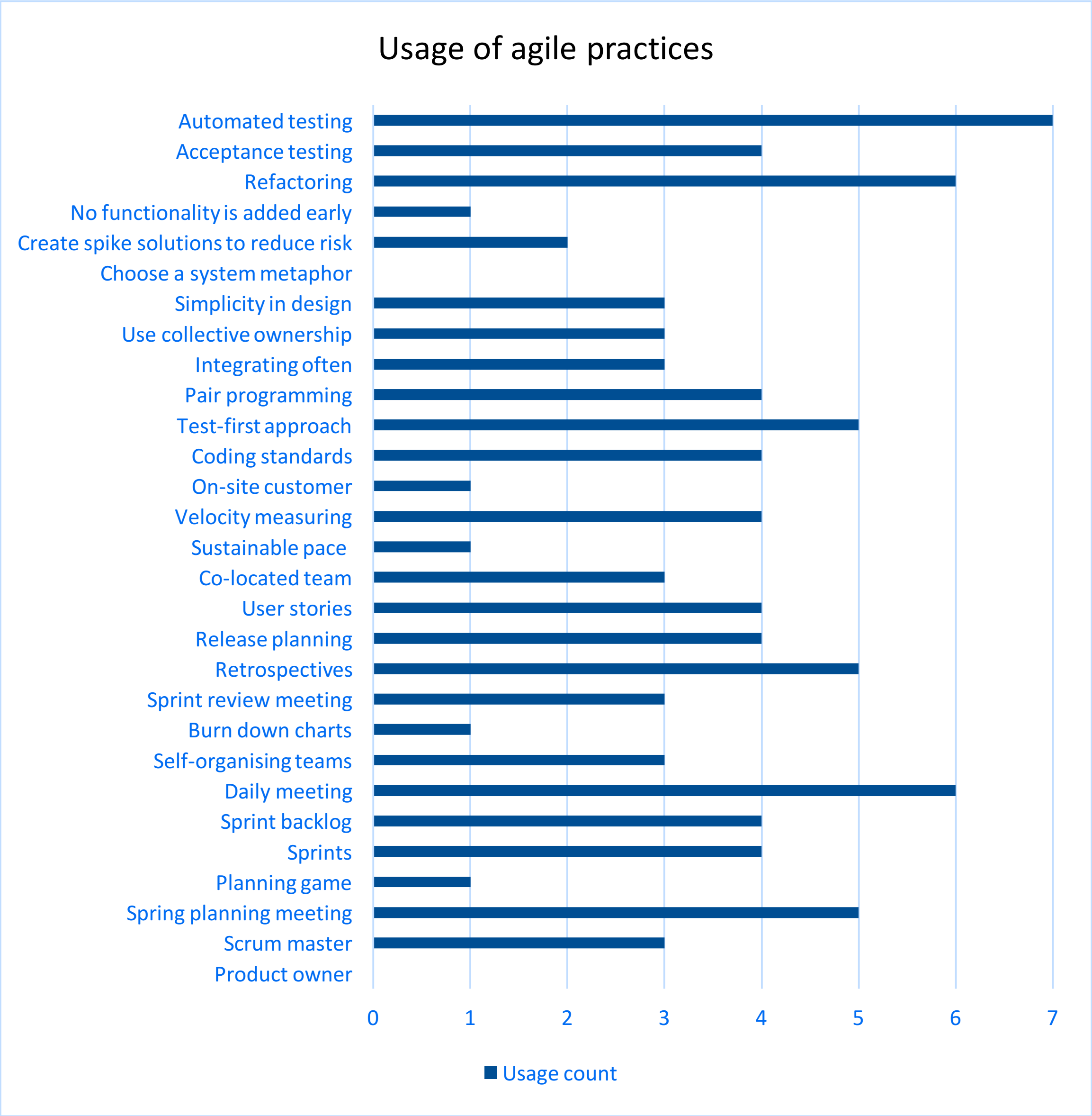


Figure 11. The use of agile practices

Looking at the least addressed topics, it can be observed that *adding no functionality early* is not a topic of interest in the literature and it does not seem to be a popular topic within the surveyed companies as well, with just 1 company using this practice. Somewhat similar pattern follows *creating spike solutions to reduce risk*, with 2 of the companies using this practice. *Velocity measuring*, however, is one of the least addressed topics within the literature, but a relevant topic in the industry with 4 of the companies using it in their everyday practice.

All surveyed companies say that they are using automated testing. The use of automated testing in SPI initiatives was reported in just 5 of the papers analysed in this thesis. This could be because automated testing has become an integral part of software processes and is not discussed anymore while improving software processes. Nevertheless, if the practice is used nowadays, it must have been used in the context of SPI in the past.

Interestingly, *product owner* is not used by any of the companies, but is a fairly talked about practice in the literature. Moreover, *choosing a system metaphor* is another practice that is not used by a single company, but is somewhat addressed in the literature with 8 papers pointing this practice out in the context of SPI.

As can be seen from Figure 12, most of the companies use continuous improvement regarding improving their software processes. This probably means the companies will not take an explicit initiative for SPI, but rather change their processes on the go, after registering the need for a change. It could also mean that processes are changed over time step-by-step adding or removing practices one-by-one, rather than overhauling the whole process.

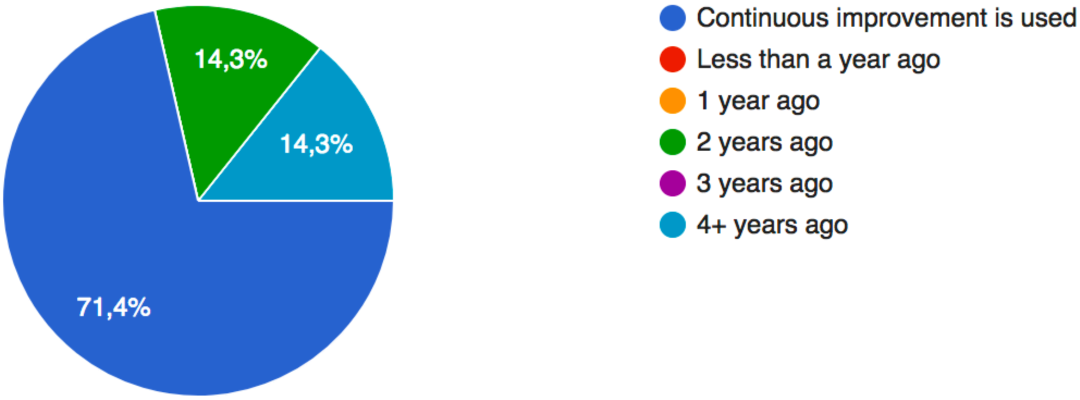
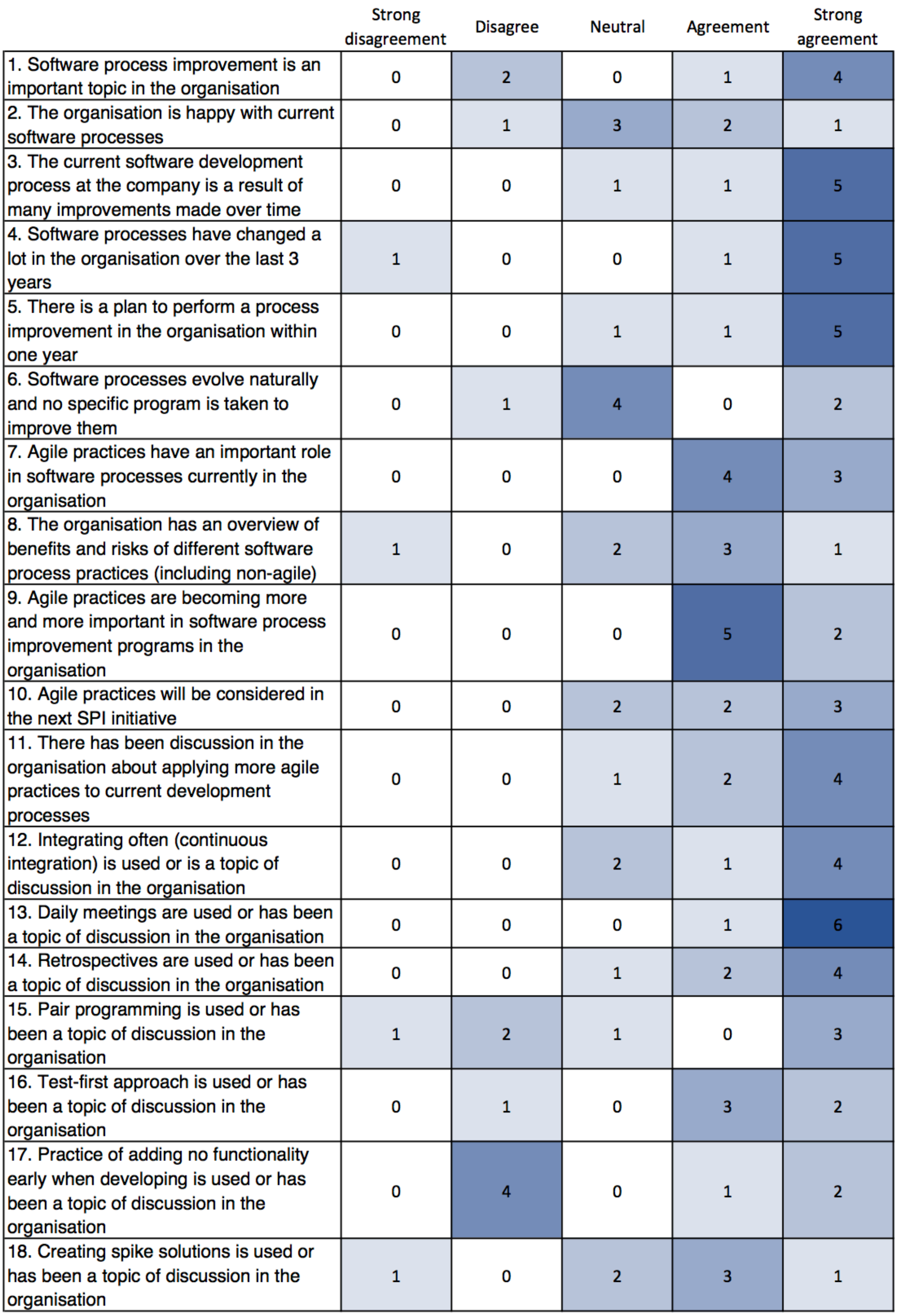


Figure 12. Last time SPI initiative was conducted in the company

When asked for the latest specific improvements made in their respective companies, one of the respondents outlined *automating tests*, while other respondent reported the introduction of *sprints* along with *sprint planning* and *planning game*. One of the companies has recently introduced *daily stand-up meetings*. One of the respondents specified that the fact that the company has multiple projects, means every project management team is a bit different and approaches the problems from a different angle, which could mean the use of different practices. 3 respondents out of 7 decided not to answer this particular question.

Last question of the survey involved rating 18 statements on a 5-point Likert scale, with 1 meaning strong disagreement agreement and 5 meaning full agreement. The results can be seen on Figure 13 with the number on the figure reflecting the count of answers in each



	Strong disagreement	Disagree	Neutral	Agreement	Strong agreement
1. Software process improvement is an important topic in the organisation	0	2	0	1	4
2. The organisation is happy with current software processes	0	1	3	2	1
3. The current software development process at the company is a result of many improvements made over time	0	0	1	1	5
4. Software processes have changed a lot in the organisation over the last 3 years	1	0	0	1	5
5. There is a plan to perform a process improvement in the organisation within one year	0	0	1	1	5
6. Software processes evolve naturally and no specific program is taken to improve them	0	1	4	0	2
7. Agile practices have an important role in software processes currently in the organisation	0	0	0	4	3
8. The organisation has an overview of benefits and risks of different software process practices (including non-agile)	1	0	2	3	1
9. Agile practices are becoming more and more important in software process improvement programs in the organisation	0	0	0	5	2
10. Agile practices will be considered in the next SPI initiative	0	0	2	2	3
11. There has been discussion in the organisation about applying more agile practices to current development processes	0	0	1	2	4
12. Integrating often (continuous integration) is used or is a topic of discussion in the organisation	0	0	2	1	4
13. Daily meetings are used or has been a topic of discussion in the organisation	0	0	0	1	6
14. Retrospectives are used or has been a topic of discussion in the organisation	0	0	1	2	4
15. Pair programming is used or has been a topic of discussion in the organisation	1	2	1	0	3
16. Test-first approach is used or has been a topic of discussion in the organisation	0	1	0	3	2
17. Practice of adding no functionality early when developing is used or has been a topic of discussion in the organisation	0	4	0	1	2
18. Creating spike solutions is used or has been a topic of discussion in the organisation	1	0	2	3	1

Figure 13. Assessed statements

box. Statements 3-5 further confirm that these companies improve their processes in a continuous manner. Out of the 7 companies 5 see SPI as an important topic. Statements 7 and 9 suggest that all of these companies see agile practices as having an important role in their processes and have the point of view that the importance of agile practices is getting even bigger in the future. Furthermore, none of the respondents suggested that no agile practices will be considered when conducting the next SPI initiative. Moreover, 4 respondents strongly agreed and none disagreed with the statement that there has been discussion in the organisation about applying more agile practices to current development processes. These aforementioned statements appear to confirm that agile practices have its place in SPI programs.

Statements 12 to 18 were designed to confirm or refute the more specific outcomes of the systematic literature review conducted in this thesis. Namely, those statements are about the most and least addressed agile practices in the literature. The differences between Figure 8 and statements 12 to 18 on Figure 13 could possibly show which of these practices has been a topic of discussion but is not practiced within the organisation. Practices like *daily meetings* and *retrospectives* both received strong agreement from most of the respondents as being used or being a topic of discussion. Even though pair programming was one of the most talked about topics in the literature, three respondents agreed that pair programming is used or has been a topic of discussion within the organisation and three disagreed, with one respondent staying neutral.

On Figure 8 two companies reported using spike solutions to reduce risk, but four companies agreed to a statement stating that spike solutions are used or have been a discussion of topic in the organisation and only one disagreed. This would suggest that spike solutions are talked about and considered for use in practice. This is somewhat contradictory to the researched literature since using spike solutions is only mentioned in one of the papers. The same can’t be said of the practice of adding no functionality early since majority of respondents said that this is not used or has not been a topic of discussion.

All in all, the survey seems to confirm that SPI is an important topic and that agile practices have an important role to play in the frame of SPI. Some of the most addressed methods in the literature were confirmed by the survey to be of relevance, but some not. The use of Scrum in Estonian organisations is high and 3 out of 4 of the most talked about practices in the literature are indeed used or talked about in those companies. Only pair programming does not seem to have the same weight in the survey as it does amongst the analysed papers. Not using pair programming is in line with the lack of usage of XP in Estonian companies. Judging by the literature, one would expect XP to be as used as Scrum, but amongst the 7 surveyed organisations, this was not the case, with only 2 using XP. However, Kanban is used by as many as Scrum, which was somewhat unexpected judging by the available literature.

3.5 Discussion

The results obtained from analysing the papers suggests that using agile practices in SPI is relevant for companies of all sizes. The published articles include examples from corporations like Ericsson and Nokia as well as small and very small companies. In not so distant past, using agile practices was deemed to be a thing for only small and medium sized companies. This is not the case anymore. Furthermore, using agile approaches does not seem to have any geographical boundaries, since examples of using agile practices in the industry come from all the major continents - Europe, Asia, Africa and both of the Americas. In general, the use of agile methods in companies of all sizes and locations indicates that agile methods have a truly global reach.

This is further confirmed by the fact that the average publication rate has steadily grown over the years after the release of Agile manifesto in 2001. Almost half (44%) of the published papers report evaluations of using agile methods in practice which would suggest that agile practices are indeed used in the frame of SPI and that it is successful, since none of the papers reported negative results. In addition, most of the papers received a high relevance score meaning they are relevant for the industry professionals who can take note of others' experiences and use them for their own good.

Using agile practices has never been strictly defined (like CMMI models for example) and that is somewhat confirmed by the lack of tools and models proposed in the literature. In the analysed papers, just 6 papers reported tools and models. This tells that utilising agile methods in SPI still needs a customised approach considering all the different factors of the organisation – the resources available, nature of the product, etc.

Looking specifically what methods and practices are used, it can be observed that the most mentioned methods are Scrum and Extreme Programming. It can't be said that this is surprising since these two methods are the most popular ones in the current field of software development. Other methods like Lean Software Development, DSDM, FDD and Crystal do get attention in several papers, but not even closely as much as Scrum and XP. For professionals looking for knowledge from the literature, the information about using Kanban, Lean, DSDM or any other less researched method in their SPI program, will not be in abundance. The use of XP and Scrum in the industry will probably continue to stay dominant, but at the same time, other methods will probably stay relevant for a minority, since the publication rate of the papers concerning the less researched methods has not significantly lowered.

The most talked about agile practices in the frame of SPI are also, understandably, practices from Scrum and XP. However, this was not entirely the case amongst 7 Estonian companies that were surveyed. It could be that for the relatively small number of respondents, it just so happened to be that the companies in this survey use Scrum. The use of Kanban was also relatively high amongst the companies. Most of the companies surveyed are small organisations. Small companies who are in the beginning of their lifespan tend to have limited resources and not so many capabilities for experimenting with their processes. The high usage of Kanban techniques could be down to that fact, since applying Kanban techniques is relatively easy and cheap. Not using pair programming could also be contributed to the bias of small companies in this survey, since pair programming can be expensive for small companies who employ just a handful of senior developers. In general, the company survey seems to confirm the importance and relevance of using agile practices in the frame of SPI. However, the use of specific methods and practices does not correspond that much to the results of the literature survey, which could be down to the factors mentioned earlier in this paragraph.

3.6 Threats to validity

In this section, different threats to the validity of the study at hand are discussed.

3.6.1 Publication bias

Publication bias is reference to the phenomenon where positive results of a study or experiment are more likely to be published than negative. There is evidence that dissemination of study results is biased. [44]

The study at hand being a literature study, it is also subject to publication bias. It implies that the analysed dataset does not include papers that report failed experiments and the lessons learned from those experiences. The bias of positive experiences can have an effect also on the conclusions drawn during the analysis. The conclusions might be somewhat positively distorted. Moreover, the results can be further influenced by the possible incompleteness of the initial dataset. Incompleteness in this case meaning that the conducted search queries did not return all the relevant results.

3.6.2 Internal validity

Conducting the study at hand involved judging every paper by its rigor and relevance and, furthermore, classifying them by their contribution and research type facets. The process of judging and classifying is subject to personal interpretation of the publications by the researchers. Personal interpretation is a somewhat subjective deciding process, meaning that two different researchers can come to different conclusions. The main internal validity threat of this study is the fact that it was conducted by a single researcher. To increase the internal validity, specific criteria for evaluating papers was outlined to able other researchers to repeat the process. The criteria followed is outlined in section 1.4. In addition, conducting the study did not involve gathering new data and all the data received was an input from the main study, which might pose as a threat to internal validity.

3.6.3 External validity

One of the threats to external validity can be the lack of knowledge of how to generalise the results. However, the main study included a wide array of publications in order to create a data set large enough for generalisation. Despite this, the study at hand inherits the limitations of the main study and its scope. To address this threat, the initial categorisation of papers being agile in the main study was revisited and corrected with more thorough approach to individual papers.

3.6.4 Different understanding of survey questions

The results obtained from the survey answers could be biased because of the nature of the presented questions. Even though the questions were thoroughly thought through and also presented to the supervisor of the thesis for validation, it could be that the conductor of the survey and the respondents understand the questions differently. It might be because of the wording of the question or just different understanding of technical terms used in the questions.

4 Conclusions

The thesis at hand set out to provide insight into the use of agile practices in software process improvement (SPI). For that cause a systematic literature review along with a survey in the industry was conducted. Conducting the systematic literature review about the role of agile practices in SPI involved classifying all the analysed papers per their contribution and research type facets, rigor and relevance. Furthermore, 35 new agile metadata attributes were defined to gain more insight into the usage of specific agile practices in the frame of SPI.

From the initial dataset of 73 papers, 59 remained for the analysis with 14 papers not analysed for reasons like not being available, not being in English or not concerning agile methods or SPI. The results of the analysed papers show Scrum and Extreme Programming (XP) as the most addressed agile methods, with Scrum being discussed in 46 and XP in 42 out of the 59 papers. None of the other agile methods like Lean Software Development, Crystal or Dynamic Systems Development were touched upon in more than 15 papers. Most researched practices in the frame of SPI were identified to be *daily meetings*, *testing first*, *pair programming*, *retrospectives* and *integrating often*. The least talked about practices include practices like *creating spike solutions to reduce risks* and *adding no functionality early*, both were addressed in just one paper. Some practices like *move people around* and *when a bug is found tests are created* were not addressed in any of the 59 analysed papers.

Several papers from all around the world provide evidence of using agile practices in SPI. These papers include evidence from small companies as well as big corporations like Ericsson and Nokia. None of the papers report negative effects of using agile methods in SPI. This could be affected by publication bias of positive results being more likely to be published. However, a survey conducted amongst Estonian software development companies of varying sizes suggests that using agile practices in SPI has a positive effect and will stay an integral part of their software development processes, while also confirming the popularity of Scrum and its practices. The survey did not confirm the high weight of XP in the literature, nor the low weight of Kanban.

Future work could involve an update study on the same topic but with an updated list of papers as an input for the analysis to see how the position of different agile methods and practices changes over time. For better generalisability of the results of the current thesis, the methods used in the systematic literature review could be repeated by other researchers. Moreover, more thorough and extensive case studies could be performed to have a more trustworthy picture of the current state of agile practices being used in SPI in the industry.

5 References

[1] Fuggetta A. Software Process: A Roadmap. *Proceedings of the on Future of Software Engineering*, 2014, pp. 1-12

[2] Kuhrmann M., Diebold P., Münch J. Software process improvement: A systematic mapping study on the state of the art. *PeerJ Computer Science*, 2016, vol. 2, no. 62, pp. 1–38.

[3] Kuhrmann M, Konopka C, Nellemann P, Diebold P, Münch J. Software process improvement: Where is the evidence? *Proceedings of the international conference of software and system process*, 2015, pp. 54–63.

[4] Kitchenham B., Charters S. Guidelines for performing systematic literature reviews in software engineering. *Technical Report EBSE-2007-01*, 2007, pp. 1-57.

[5] Petersen K., Feldt R., Mujtaba S., Mattson M. Systematic mapping studies in software engineering. *International Conference on Evaluation & Assessment in Software Engineering*, 2008, pp. 68–77.

[6] Ivarsson M., Gorschek T. A method for evaluating rigor and industrial relevance of technology evaluations. *Empirical Software Engineering*, 2011, vol. 16, pp. 365–395.

[7] Sjøberg D.I.K. et al. Conducting realistic experiments in software engineering. *Proceedings of the 2002 International Symposium on Empirical Software Engineering (ISESE'02),* 2002, pp. 17-26.

[8] Sjøberg D.I.K, Dybå T, Jorgensen M. The future of empirical methods in software engineering research. *Future of Software Engineering (FOSE)*, 2007, pp. 358-378

[9] Ivarsson M., Gorschek T. Technology transfer decision support in requirements engineering research: a systematic review of REj. *Requirements Engineering*, 2009, vol. 14, no. 3, pp. 155–175

[10] Arisholm E., Gallis H., Dybå T. and Sjøberg D. I. K. Evaluating pair programming with respect to system complexity and programmer expertise. *IEEE Transactions on Software Engineering*, 2007, vol. 33, no. 2, pp. 65-86

[11] Wieringa R., Maiden N., Mead N., Rolland C. Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requirements Engineering*, 2005, vol. 11, no. 1, pp. 102–107.

[12] Shaw M. Writing good software engineering research papers: Minitutorial. *International conference on software engineering*, 2003, pp. 726–736

[13] Sletholt M.T., Hannay J.E., Pfahl D., Langtangen H.P. What Do We Know about Scientific Software Development's Agile Practices? *Computing in Science & Engineering*, 2012, vol. 14, no. 2, pp. 24-37.

[14] Agile manifesto. http://agilemanifesto.org/ (20.03.2017)

[15] Cadle J., Yeates D. Project Management for Information Systems. London: Pearson Education. 2008.

[16] Phalnikar R., Deshpand V. S., Joshi S. D. Applying Agile Principles for Distributed Software Development, *2009 International Conference on Advanced Computer Control*, 2009, pp. 535-539.

[17] VersionOne. State of Agile Development Survey. 2017. https://www.versionone.com/pdf/state-of-agile-development-survey-ninth.pdf (20.04.2017)

[18] Boehm B. A View of 20th and 21st Century Software Engineering, *Proceedings of the 28th international conference on Software engineering,* 2006, pp. 12-29.

[19] Kupiainen E., Mäntylä M.V., Itkonen J., Using metrics in Agile and Lean Software Development – A systematic literature review of industrial studies, *Information and Software Technology,* 2015, vol. 62, pp. 143-163.

[20] Kniberg, H., Skarin, M. Kanban and Scrum: Making the Most of Both. 2010. http://www.agileinnovation.eu/wordpress/wp-content/uploads/2010/09/KanbanAndScrum_MakingTheMostOfBoth.pdf (13.05.2017)

[21] Humphrey, W. S. Managing the Software Process. Addison Wesley. 1989.

[22] Basili V., Caldiera G., Improve software quality by using knowledge and experience, *Sloan Management Review*, 1995, pp. 55-64.

[23] Basili V., Caldiera G., Rombach H. Goal question metric paradigm. *Encyclopedia of Software Engineering*, New York: John Wiley & Sons, 1994, p. 528-532.

[24] McFeeley B. IDEAL(SM): a user's guide for software process improvement. Washington: Carnegie Mellon University. 1996.

[25] Garzás J., Paulk M. C. A case study of software process improvement with cmmi-dev and scrum in spanish companies. *Journal of Software: Evolution and Process*, 2013, vol. 25, no. 12, pp. 1325‑1333.

[26] Salo O., Abrahamsson P., An iterative improvement process for agile software development, *Software Process: Improvement and Practice*, 2007, vol. 12 pp. 81–100.

[27] Aoyama M., Beyond software factories: concurrent-development process and an evolution of software process technology in Japan, *Information and Software Technology,* 1996, vol. 38, no. 3, pp. 133-143.

[28] Birkholzer T., Dickmann C., Vaupel J. A framework for systematic evaluation of process improvement priorities. *In Proceedings of the 2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2011, pp. 294–301.

[29] Esfahani H. C., Yu E., Annosi M. C. Capitalizing on empirical evidence during agile adoption. *Proceedings of the 2010 Agile Conference*, 2010, pp. 21–24.

[30] Salo O. Improving software process in agile software development projects: Results from two XP case studies. *Proceedings of the 30th EUROMICRO Conference*, 2004, pp. 310–317.

[31] Salo O., Abrahamsson P. Integrating agile software development and software process improvement: a longitudinal case study. *2005 International Symposium on Empirical Software Engineering*, 2005, pp. 187-196

[32] Qumer A., Henderson-Sellers B. A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software*, 2008, vol. 81, no. 11, pp. 1899-1919.

[33] Melo C. D. O., Cruzes D. S., Kon F., Conradi R. Interpretative case studies on agile team productivity and management. *Information and Software Technology*, 2013, vol. 55,

no. 2, pp. 412-427.

[34] Hanssen G. K., Fígri T. E. Process fusion: An industrial case study on agile software product line engineering. *Journal of Systems and Software*, 2008, vol. 81, no. 6, pp. 843-854.

[35] Sidky A., Arthur J., Bohner S. A disciplined approach to adopting agile practices: the agile adoption framework. *Innovations in Systems and Software Engineering*, 2007, vol. 3, no. 3, pp. 203-216.

[36] Silva F. S., Soares F. S. F., Peres A. L., Azevedo I. M. d., Pinto P. P., Meira S. R. d. L. A reference model for agile quality assurance: Combining agile methodologies and maturity models. *2014 9th International Conference on the Quality of Information and Communications Technology,* 2014, pp. 139-144.

[37] Fontana R. M., Meyer Jr V., Reinehr S., Malucelli A. Progressive outcomes: A framework for maturing in agile software development. *Journal of Systems and Software*, 2015, vol. 102, pp. 88-108.

[38] McCaffery F., Pikkarainen M., Richardson I. Ahaa –agile, hybrid assessment method for automotive, safety critical smes. *Proceedings of the 30th International Conference on Software Engineering*, 2008, pp. 551-560.

[39] Manhart P., Schneider K. Breaking the ice for agile development of embedded software: An industry experience report. *Proceedings. 26th International Conference on Software Engineering*, 2004, pp. 378-386.

[40] Moe N. B., Li J., Dyba T. Transition from a plan-driven process to scrum: a longitudinal case study on software quality, *Proceedings of the International Symposium on Empirical Software Engineering and Measurement*, 2010.

[41] Abdel-Hamid M. A., Abdel-Kader A. N. Process increments: An agile approach to software process improvement, *2011 Agile Conference*, 2011, pp. 195-200.

[42] Calvo-Manzano J., Garcia I., Pacheco C. Using a web-based tool to define and implement software process improvement initiatives in a small industrial setting, *IET Software*, 2010, vol. 4, no. 4, pp. 237-251.

[43] Laanti M., Salo O., Abrahamsson P. Agile methods rapidly replacing traditional methods at nokia: A survey of opinions on agile transformation. *Information and Software Technology*, 2011, vol. 53, no. 3, pp. 276-290.

[44] Song F., Parekh S., Hooper L., Loke Y. K., Ryder J., Sutton A. J., et al. Dissemination and publication of research findings: an updated review of related biases. *Health Technology Assessment,* 2010, vol. 14, no. 8.

Appendix

I. Rigor and relevance

Scoring rubric for evaluating rigor

Aspect	Strong description (1)	Medium description (0.5)	Weak description (0)
Context described	The context is described to the degree where a reader can understand and compare it to another context. This involves description of development mode, e.g., contract driven, market driven etc., development speed, e.g., short time-to- market, company maturity, e.g., start-up, market leader etc.	The context in which the study is performed is mentioned or presented in brief but not described to the degree to which a reader can understand and compare it to another context.	There appears to be no description of the context in which the evaluation is performed.
Study design described	The study design is described to the degree where a reader can understand, e.g., the variables measured, the control used, the treatments, the selection/ sampling used etc.	The study design is briefly described, e.g. "ten students did step 1, step 2 and step 3"	There appears to be no description of the design of the presented evaluation.
Validity discussed	The validity of the evaluation is discussed in detail where threats are described and measures to limit them are detailed. This also includes presenting different types of threats to validity, e.g., conclusion, internal, external and construct.	The validity of the study is mentioned but not described in detail.	There appears to be no description of any threats to validity of the evaluation.

Scoring rubric for evaluating relevance

Aspect	Contribute to relevance (1)	Do not contribute to relevance (0)

Subjects	The subjects used in the evaluation are representative of the intended users of the technology, i.e., industry professionals.	The subjects used in the evaluation are not representative of the envisioned users of the technology (practitioners). Subjects included on this level is given below: • Students • Researchers • Subject not mentioned
Context	The evaluation is performed in a setting representative of the intended usage setting, i.e., industrial setting.	The evaluation is performed in a laboratory situation or other setting not representative of a real usage situation.
Scale	The scale of the applications used in the evaluation is of realistic size, i.e., the applications are of industrial scale.	The evaluation is performed using applications of unrealistic size. Applications considered on this level is: • Down-scaled industrial • Toy example
Research method	The research method mentioned to be used in the evaluation is one that facilitates investigating real situations and that is relevant for practitioners. Research methods that are classified as contributing to relevance are listed below: • Action research • Lessons learned • Case study • Field study • Interview • Descriptive/exploratory	The research method mentioned to be used in the evaluation does not lend itself to investigate real situations. Research methods classified as not contributing to relevance are listed below: • Conceptual analysis • Conceptual analysis/mathematical • Laboratory experiment (human subject) • Laboratory experiment (software)

	survey	• Other • N/A

II. Interview questions

1 **What do you consider a software process improvement (SPI) initiative?**
Qualitative answer

2 **What software development process method is used in your organisation?**

- Agile
- Plan-driven
- Mixture
- Other (please specify)

3 **Answer this question if you use agile practices in your organisation. Please mark all agile methods that are at least partially used in your organisation.**

- Scrum
- Extreme Programming (XP)
- Lean software development (LSD)
- Crystal
- Kanban
- Other (Please specify)

4 **Answer this question if you use agile practices in your organisation. Please mark all agile practices that are at least partially used in your organisation.**

- Product owner
- Scrum master
- Sprint planning meeting
- Planning game
- Sprints
- Sprint backlog
- Daily meeting
- Self-organising teams
- Burndown charts
- Sprint review meeting
- Retrospectives
- Release planning
- User stories
- Co-located team
- Sustainable pace
- Velocity measuring
- On-site customer
- Coding standards
- Test-first approach
- Pair programming
- Integrate often
- Use collective ownership

- Simplicity in design
- Choose a system metaphor
- Create spike solutions to reduce risk
- No functionality is added early
- Refactoring
- Acceptance testing
- Automated testing
- Other (Please specify)

5 **When was the last time software process improvement initiative was conducted in your company?**

- Continuous improvement is used
- Less than a year ago
- 1 year ago
- 2 years ago
- 3 years ago
- 4+ years ago

6 **What specific improvements were made in your organisation during last software process improvement program? If any, what agile practices were included in the SPI program?**

Qualitative answer

7 **Please rate the following statements from 1 to 5, 1 meaning *strong disagreement* and 5 meaning *full agreement* and 3 meaning *no opinion/don't know/neither agreement nor no agreement***

- Software process improvement is an important topic in my organisation
- The organisation is happy with current software processes
- The current software development process at the company is a result of many improvements made over time
- Software processes have changed a lot in the organisation over the last 3 years
- There is a plan to perform a process improvement in the organisation within one year
- Software processes evolve naturally and no explicit program is taken to improve them
- Agile practices have an important role in software processes currently in the organisation
- The organisation has an overview of benefits and risks of different software process practices (including non-agile)
- Agile practices are becoming more and more important in software process improvement programs in the organisation
- Agile practices will be considered in the next SPI program
- There has been discussion in the organisation about applying more agile practices to current development processes
- Integrating often (continuous integration) is used or is a topic of discussion in the organisation
- Daily meetings are used or has been a topic of discussion in the organisation
- Retrospectives are used or has been topic of discussion in the organisation

- Pair programming is used or has been a topic of discussion in the organisation
- Test-first approach is used or has been a topic of discussion in the organisation
- Practice of adding no functionality early when developing is used or has been a topic of discussion in the organisation
- Creating spike solutions is used or has been a topic of discussion in the organisation
- Velocity measuring is used or has been a topic of discussion in the organisation

III. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Kristjan Jürisoo**,
(*author's name*)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis
On the Role of Agile Software Development Practices in Software Process Improvement,
(title of thesis)
supervised by Dietmar Pfhal, PhD,
(supervisor's name)

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **12.08.2017**