EURASIP Journal on Embedded Systems
a SpringerOpen Journal

REVIEW Open Access

Agile methods for embedded systems development - a literature review and a mapping study

Matti Kaisti[1*], Ville Rantala[1], Tapio Mujunen[1,2], Sami Hyrynsalmi[3], Kaisa Könnölä[1], Tuomas Mäkilä[1] and Teijo Lehtonen[1]

Abstract

There is a wide area of applications that use embedded systems, and the number of such systems keeps growing. The required functionality and complexity of embedded systems are also constantly increasing, and development of such products is becoming increasingly harder. This requires new thinking on the product development processes, and one such emerging philosophy is the agile methods. These methods were created by the software engineering community where they are commonly used. Since then, they have been adopted in embedded systems development; however, whether they can improve the embedded systems product development processes remains an open question. This study aims to bring forth what is known about agile methods in embedded systems development and to find out if agile practices are suitable in this domain and what evidence is there to support the findings. We conducted a literature review and a mapping study to answer these questions. The scope of this study is not only limited to embedded software development, but also to embedded hardware and integrated circuits. We have found that agile methods can be used in the embedded domain, but the methods and practices need to be adapted to suit the more constrained field of embedded product development. Furthermore, the field of embedded product development has wide diversity of products with different needs and domain-specific problems so that no single method is applicable, but rather many methods and practices are needed for different situations.

Keywords: Agile; Lean; Method; Embedded software; Embedded systems; Hardware; Integrated circuits

1 Review

1.1 Introduction

An embedded system is a specialized computer system designed for a dedicated task or a purpose which is embedded as component to a larger system usually including hardware and mechanics. There is a wide area of applications that use embedded systems from cell phones, navigation tools, video cameras, cars to appliances to name a few. The amount of functionality and complexity of embedded systems has increased substantially, making it increasingly harder to efficiently develop embedded systems products. A similar trend has been seen in software engineering where the traditional plan-based methodologies have not been able to answer to this increasing complexity and unpredictability, and as reaction to this, lightweight agile methods have gained wide popularity in the software product development. Even though embedded systems differ from the conventional software application development in many ways, there is increasing awareness in the embedded field about agile methods. However, this information is somewhat scattered on various forums, and it is additionally fairly incoherent. Development of embedded systems consists of development of software and hardware that is commonly part of a larger system or device. The purpose of this study is to bring forth the current state of agile development in the embedded systems domain and classify current academic papers to acquire the needed cohesion of the literature in a way that it is useful for both the practitioners and academics as well. The scope of the review is in the context of embedded systems where the components of

*Correspondence: mkaist@utu.fi
[1]Business and Innovation Development (BID), University of Turku, Turku 20520, Finland
Full list of author information is available at the end of the article

such systems are explicitly included in our search strategy rather than only concentrating on embedded software. As it was found in our trial searches that substantial amount of information can be found as non-academic material, we included an independent search to address this issue.

Earlier reviews [P1,P22] have found out that agile methods could be used in an embedded domain, but their use is not yet widespread. Furthermore, these studies focused on embedded software development, whereas our focus is broader. In this study, a number of studies were found that were not included in the previous reviews. In this review, also non-academic material is included. These sources generally have practical ideas on how to actually apply agile methods in the embedded domain.

This paper in organized as follows: in Section 1.2, we discuss about agile methodologies and give an overview of previous literature reviews and state the objectives of this study. In Section 1.3, we describe the research method. In Section 1.4, the results are discussed. We show what kind of studies have been done and briefly summarize the main points of the studies and discuss what is known about agile methods in the embedded world in general. Section 1.5 is a discussion about non-academic articles found by the Google search engine. The findings of this study are discussed in Section 1.6, and the paper is concluded and future directions are presented in Section 2.

1.2 Background

In this section, we start with a brief introduction to agile methods - the main idea of agile and how it relates to embedded systems development. This is followed by a summary of previous reviews in agile development, emphasizing surveys on the embedded domain after which the research questions are stated.

1.2.1 Agile methods

Plan-driven development methods were practically the only alternative for organizations until the 1990s. Royce [1] introduced a model that became known as the waterfall model in the 1970s. It has commonly been considered that the presented single-pass waterfall cycle is an exemplary development model when, in fact, Royce used it as a simplified example before proceeding to iterative models that he actually preferred.

The foundation of agile methods is in the iterative and incremental development. Agile methods have gained increasing popularity since the 1990s when the agile movement begun and several software production processes evolved such as the well-known Extreme Programming (XP) and Scrum methods. The teams are commonly small, self-organizing and co-located, working closely together that helps in producing a high-quality software. Frequent feedback from closely collaborating customers is also promoted as a means of fulfilling customer needs [2].

The agile methods are a set of practices created by experienced software developers. The agile methods can be seen as a response to plan-driven processes that emphasize extensive planning and documentation with strict processes, and have a view that specifiable and predictable solutions to problems exist [3]. In contrast to plan-driven methods, Agile Manifesto [4] highlights the following: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and responding to a change over following a plan. Even though the manifesto highlights the values of the items on the left more than the ones on the right, it does not abandon the ones on the right either. The main differences between the plan-driven and the agile development paradigms are illustrated in Figure 1. In plan-driven development, there are consequent phases which are usually handled by phase-specific teams. Information between the phases is transferred using extensive documentation. In agile methodologies, the development is done in incremental iterations in integrated teams. Requirements for the developed system are stored in the product backlog.

Besides the manifesto, there is no single and exact definition of agile, but rather there are various methods that share the same philosophy which call themselves agile. Most methods have iterations with continuous refinement of plans and goals. However, each method has its own practices and terminology, putting emphasis on different issues in software development, and hence can be adopted to suite for different situations.

For example, XP is a set of pragmatic practices that emphasize extensive testing, code revise and pair programming [5], whereas Scrum is a development framework that focuses on management issues in projects where planning is difficult. The core of Scrum is in frequent feedback loops and Sprints that take place in daily stand-up meetings and with monthly (or more frequent) planning meetings [6]. Another popularity-gaining methodology, Kanban, differs from the previous as only having few practices and promoting gradual change and a constant flow over timeboxed iterations [7].

In this paper, the methodological analysis is based on the division between the plan-driven and the agile paradigms. It has been argued that other methodological paradigms also exist. For example, open source development has been seen as a distinguished development approach. On the other hand, the open source development can be seen as one of the many development domains where agile methods can be successfully applied [8].

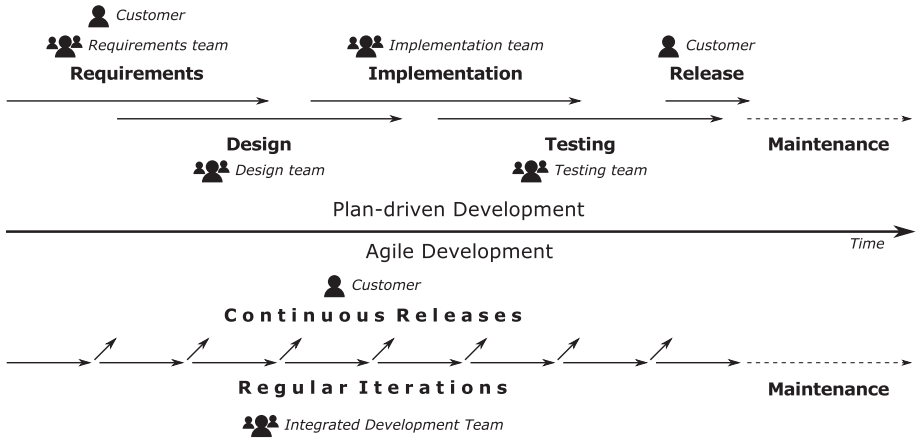

Figure 1 The main differences between the plan-driven and the agile development methodologies.

1.2.2 Review questions

In this review, the main objective was to find out the current state of agile methods in the field of embedded systems development. We made a clear distinction between *embedded software* and *embedded systems* development. In our discussion, the former is software development which is constrained by the used hardware, whereas the latter is not only constrained by hardware but also includes hardware development.

As the purpose of this study was to find out what is currently known about agile methods in embedded systems development and if these methods are suitable in this domain, we focused on the following questions:

RQ1 What is currently known about agile methods in embedded systems and embedded software development?
RQ2 Are agile methods suitable for embedded systems and embedded software development?
RQ3 What kind of evidence is there to support these findings?

1.2.3 Summary of previous reviews

The first found review about agile methods in general was published by Abrahamsson et al. [9] in 2002, reviewing the existing agile methods and practices. Dybå and Dingsøur have written a review on the existing empirical studies in agile software development [3].

There have also been systematic literature reviews on agile methods in embedded systems development prior to this study [P1,P22] which are analyzed in more detail in Section 1.4.1. Albuquerque et al. [P1,P22] present a systematic literature review, which is similar to our survey, on agile methods in embedded systems development. In the review, the current state of agile methods is examined; the most often used agile method is found, and the challenges in adopting these methods in embedded systems are discussed. The authors point out that the suitable agile methods for embedded systems development should be surveyed in more detail.

Even though the starting point of Albuquerque et al. [P1] is close to our survey, the results, to some extent, differ. Our approach is more hardware development oriented, and we have emphasized embedded hardware and integrated circuit development in our review. Albuquerque et al. have included a total of 23 articles from which 12 articles are also selected to this study. The difference in the found articles is due to slightly different viewpoints in the article selection process and in the inclusion and exclusion criteria. Albuquerque et al. emphasize agile methods, such as Scrum or XP, while our study concentrates on agile and lean methods putting more emphasis in the actual agile hardware development including integrated circuits.

The literature study by Shen et al. [P22] concentrates on studies about the usage of agile methods in embedded software development. The emphasis in the selected articles is in the application of agile principles. The survey includes 40 articles. Regardless of the slightly different approach, there are 12 same articles in these surveys where the difference is mostly due to different search strategies. The observation made by Albuquerque et al. is shared in that a more rigorous research is needed.

In our study, we did not limit the review to any particular agile method and we also included lean methods as an important part of the review. We also decided to include several search strings that emphasize hardware and not just embedded systems to make the search more comprehensive. The development of embedded hardware and integrated circuits is included because of their essential role in the development of embedded systems. This study also includes non-scientific forums which were found to be a relevant source of information on agile methods.

1.3 Research process

A systematic literature study is a systematic and repeatable approach to identify and study all relevant evidences, i.e. primary studies, on a specific research question or phenomenon [10]. The method consists of literature search, study selection, data extraction and synthesis. Systematic literature studies can be roughly divided in two categories [10]: systematic literature review (SLR) focuses on finding existing evidence on a specific question, while systematic mapping study (SMS) categorizes existing studies on the topic in order to show the gaps of knowledge.

In this paper, we performed a study that incorporates features on both SMS and SLR on the use of agile software development methodologies in the context of embedded software and embedded systems development. We decided to map the existing evidence and, thus, to identify gaps in current research in addition to synthesizing the found primary papers.

We use major publication databases and search engines available at our university. We started the search process by conducting several pilot searches using different search terms and search options and decided to concentrate on the terms 'agile' and 'lean' from the process method viewpoint and to the terms 'embedded software', 'embedded system', 'hardware' and 'integrated circuit' from the viewpoint of domain knowledge. Thus, we used the following search string: `[agile AND ('embedded software' OR 'embedded system' OR 'embedded systems' OR hardware OR 'integrated circuit' OR 'integrated circuits')] OR[lean AND('embedded software' OR 'embedded system' OR 'embedded systems' OR 'integrated circuit' OR 'integrated circuits')]`. The search was performed using full text option, when it was available in the search engine. The searches were done in December 2012.

A study was selected if it was from the field of agile development of embedded systems, embedded software, electronics hardware or integrated circuits. We included expert opinions, lessons learned papers and articles presenting empirical results from students, professional software developers and academics. We did not include editorials, book reviews, or interviews. Papers documenting the same original work were considered as duplicates, and therefore, only one of them was included. Only articles written in English were included.

The article search was divided into three stages as shown in Figure 2. In the first stage, the search engines were divided between three authors and the studies were included based on the title of an article. An article was selected if the article was from the field of agile development of embedded systems, embedded software, electronics hardware or integrated circuits. From 20,430 hits, only 379 were selected (most of the duplicates removed) for the second stage.

In the second stage, the abstract of each selected paper was read by two authors. By using the same criteria as in the first stage, the two authors voted on each article to include it or not for the third stage. The authors agreed in about 80% of cases. In a case where one author suggested inclusion and the other did not, a third opinion was used to decide. After the second stage, 58 articles remained to be analysed in the third stage.

In the third stage, the articles were randomly distributed to three authors so that each article was skimmed through by at least two authors. In this stage, the papers were checked in detail in terms of the inclusion and exclusion criteria. In the end of the third stage,a total of 28 articles were selected for the final review where the articles were read carefully and data was extracted from them.

In the data extraction, the articles were categorized in terms of their content and type. The categories were pre-defined, and each article was classified by at least two authors. From the content, we divided the studies to be either embedded systems or embedded software related. In our analysis, a 'systems article' is defined to have some content on hardware or mechanical development, and a 'software article' is about embedded software development possibly with embedded systems-related constraints. In addition, articles were categorized being qualitative, quantitative or neither. Also, the used agile method from each article was mapped. The results from the first extraction phase are collected in Table 1.

After the categorization, the selected studies were read carefully and they were summarized. We used descriptive synthesis to analyse the findings. We identified three themes - method development, adoption of agile methods and experience reports - that recurred in different papers, and we classified the articles based on these categories.

1.4 Results

The 28 chosen articles were classified in terms of their content and type as described. The selected articles have been published quite recently. The first published paper was in 2003. The amount of published studies per year has remained fairly stable as shown in Figure 3.

The selected articles have several different approaches in studying agile methods. Some of them focus on previously known methodologies, such as Scrum or XP, while others concentrate on the agile practices and the original Agile Manifesto [4] without utilizing any certain methodology. There are also papers focusing on development of new agile methods. The distribution of different agile methods appearing in the selected papers is illustrated in Figure 4. The figure shows that XP is the most frequently mentioned method. From the more detailed information,

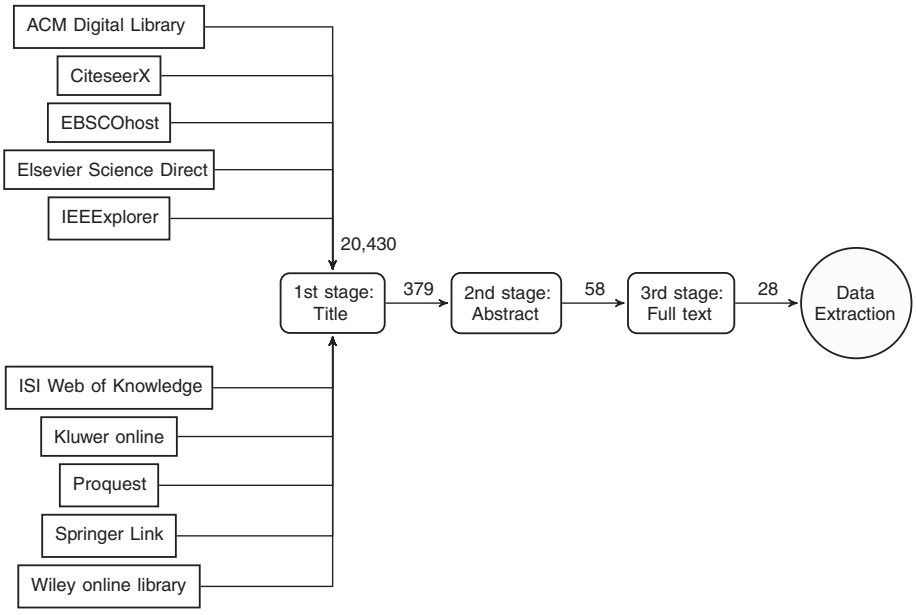

Figure 2 Illustration of primary study selection process and used search engines. The numbers of articles selected for each stage are shown.

presented in Table 1, one can notice that in many studies, XP and Scrum methodologies are used together. The *Applied agile* category includes studies where the used method is unique and typically based or modified from previously known methods such as Scrum or XP. The category *other* includes methodologies such as test-driven development (TDD) and agile manufacturing. All of the different methods appearing in the other category are named in Table 1. The final method category is for studies which include *multiple* agile methods. These articles are typically reviews which survey the previous works on the field.

From the selected articles, 18 included a case study or an experiment report. Project-specific characteristics are presented in Table 2. The projects and teams were quite often small, but the specific team size was left unmentioned in half of them. Few of the articles concentrated on large organization-related challenges. Quite many projects had changing requirements, which had led to selection of agile methods. Project areas were quite wide in embedded system and software development, e.g. satellites, telecommunication and processor firmwares. In the majority of the articles, the success of a project was based on the impression of the authors. In six cases, there were also some measurements based on which success was defined: effective code lines per hour [P3,P26], defect rates monitored [P26,P27], time used [P2,P26], failures in field and upload success rate [P13], performance measurements compared to previous similar products [P20] or test coverage [P3]. One case [P3] used a survey to get insight about how people felt.

1.4.1 Review articles

Among the selected articles, there are three studies which are considered as surveys. These articles include two systematic literature reviews [P1,P22], and they were briefly presented in Section 1.2.3. Albuquerque et al. [P1] present a systematic literature review on the agile methods in embedded systems development. The authors in [P1] have found out that agile methods have had a positive impact on embedded systems development while their use is still not widespread. They present potential research lines which should be investigated in more detail. Research should be conducted on suitability of different agile methods to the development of embedded systems and how these methods affect the product quality as well as to the development cost and effort. They also point out a need for investigation on how to adopt agile methods when safety intensive requirements have to be taken into account. Albuquerque et al. have made a conclusion that the suitable methods for embedded systems development should be surveyed in more detail.

The other extensive SLR among the selected studies is authored by Shen et al. [P22]. It surveys the studies on agile methods of embedded software development. The selected articles are surveyed especially in terms of the application of the agile principles. Shen et al. share the observation of Albuquerque et al. that more rigorous research of the field is needed. They have noted that XP and Scrum are the two most used agile methods in the field of embedded software and see that the characteristics of embedded software development bring new challenges into applying these methods. The authors see that the

Table 1 Article content

	Embedded		Analysis		Agile Method				
Number	**Software**	**Systems**	**Qualitative**	**Quantitative**	**Scrum**	**XP**	**Applied**	**Other**	**Number**
P1		x	x	x					P1
P2	x		x		x	x		Plaform-based design	P2
P3	x		x	x		x			P3
P4	x		x			x		TDD	P4
P5	x		x		x	x			P5
P6	x		x			x			P6
P7	x		x		x				P7
P8		x	x				x		P8
P9	x		x					Agile manufacturing	P9
P10	x		x						P10
P11	x		x						P11
P12	x							DDD	P12
P13	x		x	x			x		P13
P14	x		x			x			P14
P15		x	x			x	x		P15
P16	x		x				x		P16
P17	x		x	x	x	x			P17
P18	x		x				x		P18
P19	x		x		x	x			P19
P20		x	x					HW/SW co-design	P20
P21		x	x		x				P21
P22	x			x					P22
P23	x	x	x					TDD	P23
P24	x					x		TDD	P24
P25		x	x				x		P25
P26	x		x			x			P26
P27		x	x			x			P27
P28		x	x				x		P28
Total	20	9	25	5	6	11	7		

DDD, document-driven development; TDD, test-driven development.

investigation should be extended to a wider scope outside the well-known XP and Scrum methods. They also conclude that active collaboration with embedded industry is essential in the research of agile methods for embedded software development.

While surveying the used agile methods for embedded systems development, Srinivasan et al. [P25] have noted that there are both technical and organizational issues in adoption of the agile practices. They noticed that the main gap in the literature is the absence of reports on failures in the agile adoption in embedded systems development. Based on the published literature, the authors made recommendations on how the focal principles, such as test-driven development or requirements management, should be utilized to evade the issues. They, for instance, recommend to tailor the agile practices into larger organizational context and harmonize the requirement management to support modifiability, maintainability and dependability. They conclude that adoption of agile methods needs organizational support. Willingness for organizational culture change is required especially because of the soft factors present in the agile practices.

1.4.2 Method development

The selected articles include three studies that propose a new development method for embedded software [P2,P24] and systems development [P12]. Smith et al. [P24] propose a method that emphasizes test-driven

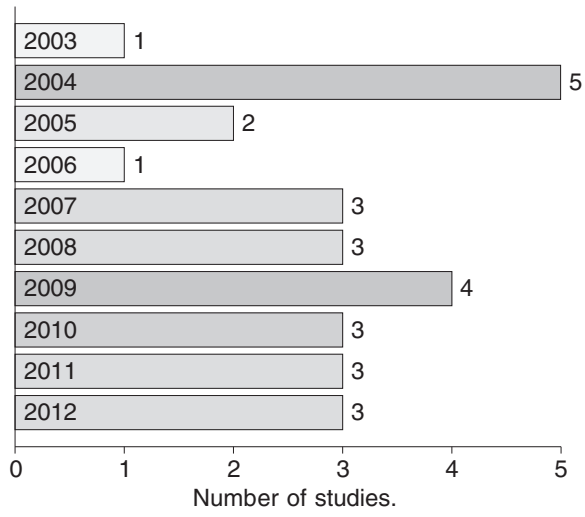

Figure 3 Year of publication.

approach that could be considered most suitable for embedded software development. Cordeiro and Barreto [P2], in turn, propose a platform-based design method where a platform consisting of frequently used functionality can easily be extended and modified for different projects. The method proposed in [P12] might be better suited for embedded systems design as it is based on an appropriate documentation system that simplifies information sharing between stakeholders. Four other studies can be considered as method development, which do not propose a new method, but rather extensions to existing ones. In [P11], various pragmatic suggestions are proposed on how large-scale organizations can better utilize agile methods. Articles [P9] and [P10] compare agile methods to other methods, and in [P28], agile methods are introduced into requirements engineering, and it is tested by a case study.

Smith et al. [P24] propose a development process where the authors try to find the most suitable practices from test-oriented processes that can be adapted to the embedded software development. The proposed development method, to some extent, lies on the findings of an experience report by Smith et al. [P23] where undergraduate students experimented with test-driven development in embedded domain. The core of this new embedded test-driven development approach is a subset of practices from XP. The practices that were modified to the embedded domain include test-driven development, refactoring, simple design, pair programming and continuous integration. To implement test-driven development in the embedded domain successfully, some changes are required to the practices. For example, refactoring should be interpreted as making improvements in speed and in lower power consumption even if this might reduce the portability, clarity and modularity. The proposed method addresses the issue of different life cycles in embedded products compared with conventional software applications and proposes how this difference could be addressed in product development.

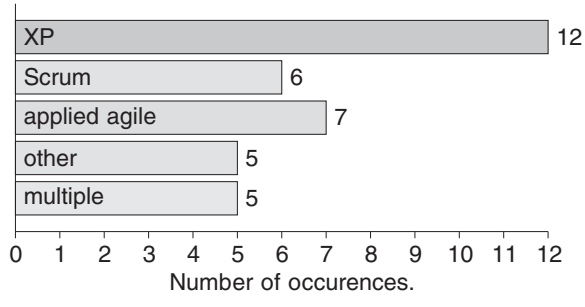

Figure 4 Used agile methods.

In [P28], Waldmann points out that in agile, requirements engineering process should be transparent to all stakeholders. In the presented case study, agile practices had been applied in requirements engineering of a new generation of platform components (generic components that can be used over different products) for hearing solutions. It was found that the same product should be customizable to answer the needs of different customers instead of developing different products for all the different customers. A similar view is found in [P2] where Cordeiro and Barreto propose a platform-based design development method where a microprocessor-based architecture can be rapidly extended, customized for a range of applications and quickly delivered to customers. Most of the used practices in the proposed method are adopted from XP and Scrum methods. Cordeiro and Barreto also briefly discuss the results of the proposed method, applying it in three small projects with one to four developers in a project with two to three sprints. It is argued that the proposed method showed a reduction of development time in the case studies, but it is acknowledged that development methods are difficult to compare.

In [P12], Luqi et al. introduce a new document-driven development (DDD) method. They point out the importance of documentation especially in complex real-time systems and emphasize the use of appropriate documentation. The core of the proposed method is in the document repository where all information concerning the development is stored. Information can be represented in different forms, depending on who needs the information and thus providing the means of communication between stakeholders, e.g. managers, developers, sponsors, maintainers and end-users. The aim is also to use the DDD to integrate all used methods and tools into this new system,

Table 2 Case studies and experiment reports

Number	Method(practices)	Team size	Success	Discipline	Project characteristics
P2	A new platform-based design method based on XP, Scrum and agile patterns (e.g. Sprint and Product backlogs)	1-4	The development platform reduced substantially development time of the product	Three embedded system projects: pulse oximeter, digital soft-starter and the induction motor simulator equipments	Team uses pre-designed and pre-characterized components instead of full custom design methods
P3	XP (The planning game, short development cycles, pair programming, test first programming, collective code ownership, frequent integration, never solving a problem that has not occurred, refactoring, minimal documentation)	Total 29 people in 4 teams	XP can be used to develop complex mission-critical systems after the team starts to work	Software development of mission-critical two-way radio systems projects	There were four about 18-month projects, where requirements change
P4	XP practices (unit test first with mock objects, automate building software and testing, write code in testable fashion)	N/A	Agile techniques can be used also in embedded development	Software for control boards of automated guided vehicles	The team had used agile in web and desktop applications previously
P5	Selected practices from Scrum (Sprint, Sprint planning meeting, Daily scrum, Sprint review/ Retrospective) and XP (simple design, unit testing, refactoring, pair programming, collective code ownership, continuous integration, on-site customer, sustainable pace, coding standards)	7	Experiences are mostly positive, but full adoption is seen a slow process	Firmware for processors	The team members have specialized domain knowledge, the requirements change and hardware dependencies force change. Team is also distributed
P6	XP (planning, small releases, metaphor, simplicity, test, refactoring, pair programming, common code ownership, continuous integration, 45-h week, on-site customer, coding standards)	N/A	In small teams, development cycles were shorter, but in the whole project, the results were not satisfactory	Telecommunication software development	The specifications and requirements are fuzzy in the beginning of project and change during the project
P7	A new piloting method when introducing Scrum to the company	20 and 15	Team can be Agile, even though the company is not and piloting can reduce resistance to change	Software and hardware solutions for the wireless and automotive industries.	A large, distributed company
P8	Agile manifesto-based method (individuals and interactions, emphasis on working system, collaborative interface with sponsor, responding to change)	N/A	Project was successfully finished within short schedule and limited time	Design and development of two small satellites	The project has high level requirements of uncertainty, cost and schedule
P12	Document-driven development (DDD)	N/A	Improvement was seen in more effective design procedure and improved communication between all stakeholders	Software for a system to administer therapeutic intravenous fluids	The deployment is for long periods of time and used globally. There are mission-critical requirements; frequent changes and components are developed in different organizations with variety of stakeholders

Table 2 Case studies and experiment reports *(Continued)*

P13	Agile practices (unit test, test first)	N/A	Agile elements are serious options for development of embedded software	Embedded software for busses and coaches	High degree of customer-specific functionality to be implemented, e.g. within days
P15	XP and Agile (user stories, start simple, user involvement, refactor)	N/A	Project was viewed successful both internally and externally	A system for scheduling satellite tracking stations	The developed system is large, and the organization complex. The development team is inexperienced, and there have been previous failures in the project using waterfall
P18	Agile in general (e.g. Boehm's spiral model, common room, pair programming)	4	Successfully created first software versions in schedule despite some requirement changes	A control software for satellite camera equipment	The requirements change, and the schedule is tight. Engineers share time with hardware development
P19	Scrum and XP (e.g. user stories)	N/A	In large organizations, agile development requires very skilled developers and has to be a combination of old and new practices	Telecommunication	There are challenges in work allocation between teams in large organizations when using agile. Requirements engineering is also used in this project
P20	HW-SW codesign (hardware simulation and code analysis tools)	N/A	Higher performance was seen relative to conventional HPC design	A high performance computing system	There are power and performance challenges
P21	Scrum (sprints, daily stand-ups, sprint backlogs, sprint review and sprint retrospective and customer involvement)	15 people in 4 teams	Agility was applied only to software team, and to fully benefit from it, it should be broadened and deepened	Advanced communication system project	The original waterfall project suffered from frequent requirement changes, lack of communication and expensive overheads
P23	TDD-based tool and method	N/A	Current unit testing tools can be adapted to embedded environment	Experience on a tool in digital signal processing applications	Experience on embedded test-driven development tool usage in university laboratory course with undergraduate students
P26	XP (e.g. collective code ownership, unit tests, always enabled tracing system)	4-6	Unexperienced team's productivity cannot be distinguished from best teams in industry, when the team uses Agile practices	Software for a grain monitor	The software developer team is unexperienced and small
P27	XP testing techniques (always on trouble log, dual-targeting, unit tests, domain level tests)	About 5	Amazingly low bug rate and it was easy to distinguish whether the problem is in software or hardware	Software for a mobile spectrometer	A new product is developed from a scratch with changing hardware
P28	Agile requirements engineering	Over 100 in the project	Flexible requirements engineering provides business value, even with severe resource constraints	Hearing solutions based on platforms	The new platform project is large with inherited functionality and professional requirements engineers included for the first time

but how this could be achieved remains an open question for the researchers.

Articles [P9] and [P10] compare agile to other methods. In [P9], Kettunen compares agile manufacturing and agile software development models and also finds out new manufacturing concepts that could potentially be adopted in software production. An example case of network element products also proves the similarities between these two methods. In [P10], Kettunen and Laanti compare eight different software process models from which three are agile methods, namely feature-driven development, adaptive software development and XP. The analysis results can be used for selection of a suitable model based on the anticipated project problems. The validation of the results is done through four examples based on certain past real-life projects. Kettunen and Laanti [P11] also have various pragmatic suggestions on how agile methods could be used in large-scale embedded software products based on industrial experience. It is noted that in a large-scale new product development organization, it is important to have a comprehensive view of the whole organization. One simple suggestion on creating a more efficient work-flow is to co-locate related hardware and software developers. It is pointed out that in a large company to work in an agile manner, it is not enough to concentrate only on teams and projects, as is usually done in agile methods. Furthermore, a company should understand what are the goals it tries to achieve with agile methods and, hence, what kind of methods or practices are needed. The answer is company dependent, and not all companies should adopt the same ways.

1.4.3 Adoption of agile development methods

Only four studies [P4,P5,P13,P14] have addressed the introduction of agile development methods in the embedded domain. However, all of them are expert reports from different projects in the industry; [P4,P5,P13] reported a case while [P14] compared experiences from previous projects to the XP's practices. All these studies focused mainly on the embedded software development. None of the articles included into this review addressed the question of introducing agile methods in embedded systems development.

The adoption of agile methods in a new domain has raised arguments. On one hand, Matthews [P14] argues that the XP's practices are not agile nor software development specific; instead, they are the baseline for every practical working method. Matthews acknowledged that agile methods will be adopted in the embedded world in the near future; however, he demands consideration on which practices should be adopted. Also, [P5,P13] noted that an agile method should not be followed dogmatically in the new domain. On the other hand, Fletcher et al. [P4] remark that there was nothing which would have prevented them from using the same methods that were used in the conventional software development.

Only a few disadvantages were reported. In [P4], the authors raise the lack of support of the software tools to utilize agile practices as one of the challenges. Mannhart and Schneider [P13] and Greene [P5] argued that the importance of domain knowledge in the embedded software development hinders the use of the principle of shared responsibilities.

Drivers for agile adoption varied: Constant changes required by the hardware team were mentioned as one of the main reasons in [P5]. In [P13], the main focus was to clarify the process on adding specialized functionalities and individualizations asked by the customers to the buses with embedded software. In [P4], the team decided to use the same methods that they had been using previously. All studies judged that adoptions were successful - although one engineer left the team arguing that they 'were not acting enough as a team' in the case of [P5].

Similarly, the ways of introducing agile differed. In the case of [P5], no formal education was reported before the adoption of Scrum and XP - the members of the team read through a book of XP beforehand. In [P4], a team that had a long experience on XP, in other domains, was hired to work with embedded software. However, their report focuses on introducing test-first practices to the embedded domain. In [P13], the agile methods were introduced to the process step by step: in the first phase, they presented only test-first and unit-testing practices to embedded software teams. They used also Goal-Question-Metrics approach (see e.g. [11]) to foster the adoption; however, no results were reported as the project was still ongoing.

1.4.4 Experience reports and case studies

Experience reports and case studies discuss either techniques used for improving quality and predictability early in the development process, or reasons and practices behind successful adoption of agile methods in embedded systems development.

In [P27], Van Schooenderwoert and Morsicato introduce how using domain-level simulations helped in isolating bugs easier while developing a mobile spectrometer, whereas in [P20], Shalf et al. present how performance and power consumption of a high-performance computation application could be estimated by simulating the system model in different abstraction levels. In both articles, it is also described how a simulation environment was developed to enable and support these approaches.

A few articles mentioned the importance of adaptation of agile. For example, when applying XP into telecommunication software development [P6] and mission-critical two-way radio systems [P3] development, some sort of top-level documentation is needed which is not pointed

out in XP practices [P3,P6]. In [P6], Gul et al. also noted that applying XP in small teams resulted in shorter development cycles, but applying XP to the whole project did not give satisfactory results. In [P3], Drobka et al. suggest using strong outsider coach, training, periodic audits and code spot-checks.

Also, articles [P8,P15,P21] emphasize the importance of process tailoring to get the best out of agile practices in embedded systems development. In their experience report [P8], Huang et al. describe how the organisational structure and process for developing high technology satellites could be modified according to the agile principles to meet cost and schedule requirements. As Morgan [P15] points out, the process tailoring can also mean that the development team itself can operate in an agile manner, whereas it has to adapt working with non-agile teams. In a government-funded development project of a system for scheduling satellite tracking stations, the team had to create some design artefacts that were used only externally. Shatil et al. [P21] bring forth several topics that were considered important during the agile adoption process. Involving management in early stage of the adoption process enabled multidisciplinary teams to be familiarized with the software team's process and help other teams to be more synchronized with the software team. The project management also started to manage the project by taking into account the software team's iterations. As a means of getting members of the development team committed to the adoption process, it described how their feedback was used to find main concerns and solving these. Finally, agile practices taken in to use were tailored to find the best fit for the development environment.

Instead of adapting some specific agile method, [P18] and [P26] used some agile practices in development. In [P18], dos Santos Jr. et al. describe the usage of an iterative model with agile characteristics in a small team that developed a control software for a satellite camera equipment. Creation of useful and reliable software rapidly was achieved by reacting to changes, using pair programming especially in complex routines, strong communication and allowing developers to make most technical decisions.

In [P26], Van Schooenderwoert describes a novice development team challenging an experienced team where the performance of the teams was measured by several industry standards. The agile practices behind the success were collective code ownership and strong unit tests. The lack of experience of some team members was overcome by agile software development techniques and the presence of senior level developers that allowed knowledge transfer between teams.

Studies presented in [P7] and [P19] focus on agile methods in large organizations. In [P7], Heidenberg et al. suggest that piloting is a good way to overcome resistance to change and to convince management that a team can be agile even though the rest of the company is not. A three-step piloting method (marketing the pilot, preparing the pilot and executing the pilot) is introduced and validated as a case study in two pilots within the same company. In [P19], Savolainen et al. describe challenges of large organizations in embedded systems when transitioning into using agile process models. Using user stories may present a problem in large embedded systems, since user interaction might be far from the implemented framework. Use of Scrum with key requirements engineering practices is seen a good way to introduce agile methods to embedded systems. Savolainen et al. also conclude that it is a good idea to preserve some of the key practices instead of starting from scratch when introducing agile methods.

In [P16], Ronkainen and Abrahamsson analyse prospects of using agile methods in embedded software development under hardware constraints based on observations in signal processing application development. They point out that agile methods are not targeted for developing embedded software and that new methods for embedded software development need to solve many challenges: real-time constraints of the hardware, need to experiment the software on hardware and need for documentation so that all stakeholders stay informed and that the development is inherently test-driven with hardware-related constraints. The authors conclude that agile methods might offer solutions for embedded software development, but the methods need to concentrate on the embedded domain-specific requirements.

Salo et al. [P17] have arranged a questionnaire of the actual use and usefulness of XP and Scrum in organizations developing embedded software. The survey involved 35 individual software development projects and focused on observing the level of use as well as the experienced or expected usefulness of the agile methods. The questionnaire concentrated to XP and Scrum and the separate practices involved in these two methods. Authors have asked which methods have been utilized, if any. They also asked how frequently the separate practices of Scrum and XP were utilized and have they been found useful. The results show that at least two thirds of the respondents have utilized one or both of the methods. From these two methods, XP was used more in the investigated companies. Those familiar with the methods have found them mostly useful. However, the authors point out that no broad generalizations can be made from the results.

1.5 Non-academic material

In order to review non-academic material available, a Google search was conducted on January 16, 2013 using the same search terms as for the literature review. Since it was assumed that plural forms of search terms should be automatically included by the Google search engine, they were excluded from the search. This resulted in seven

distinct search strings with 50 first results from each taken for further analysis. Following the procedure used in the literature review, search results were then split between two authors and non-relevant search results having little or no content discussing agile or lean were excluded. In the next step, two authors read through all 48 search results that passed the previous step and voted for inclusion or exclusion of each with a third opinion used when two authors did not agree. After this step, 21 search results were left for deeper analysis. Search results were excluded if there was no agile or lean content, they were already covered by the SLR, or they were duplicates pointing to a same web page. It is notable that searches done using search term 'lean' resulted only in three results passing through all phases. Even though the 14 of the selected search results were from the recent 3 years (2010, 2011, 2012), also some older ones were found starting from 2004. We have included the most relevant sources concerning our study in the references.

The results can be categorized into two groups by the author type: one group was those whose work was included in the literature review (e.g. Michael Karlesky, Pekka Abrahamsson, Petri Kettunen and Nancy Van Schooenderwoert), while the other group consists of people contributing to agile embedded HW/SW development mainly by maintaining blogs, writing articles to electronic newsletters, providing training and/or consulting and giving speeches in agile conferences (e.g. Zubin Irani, Neil Johnson and Timo Punkka).

Instead of bringing new and revolutionary ideas, search results had generally a practical approach in applying agile practices into embedded systems design. A couple of the search results provided information about where to find more information about agile in embedded software development, or whom to ask for more information [12,13]. Some considered the characteristics of embedded systems that should be taken into account when adopting agility, and others gave practical advice on useful agile practices in embedded systems design [14]. Applying agile methods to hardware development was also discussed in several sites [15-18].

It is also noted that even though hardware developers should look for the common practices in the software development domain when adopting agile practices, the characteristics of hardware development should be taken into account and practices adapted according to those [17,19-21]. For example, delivering a working prototype in the end of each iteration is not possible. The key for hardware developers is to resist getting caught up with the differences between software and hardware and to instead focus on the similarities [17,18].

Common themes found involve use of test-driven development, continuous integration, dual targeting, iterative development and customer collaboration [16,17,22,23]. Dual targeting answers to the lack of prototypes in the end of every iteration, but it also brings carefully thought design to the software-hardware interface [17,21]. Continuous integration including automatic testing will help to identify failures in the early stage [16,17,23]. There are several ways to improve customer collaboration, e.g. user stories can be used, but the term user can also refer to nearer customers, not necessarily to the end user [21].

1.6 Discussion

In this paper, we studied what is currently known about using agile methods in the development of embedded systems and embedded software (RQ1). The results showed that the research is rather scattered and mainly driven by industry reports. It was found that there is no one method for the diverse world of embedded systems development, but many emphasize different viewpoints. There were multiple new development methods proposed that would better suit the needs and constraints of the embedded domain in addition to ideas on how to scale agile methods in large organizations and in ways agile methods have been adapted to suit various needs of different companies.

Our second research question was whether the agile methods are suitable for the development of embedded systems and embedded software. For example, Ronkainen and Abrahamsson [P16] lay out requirements for agile methods that need to be addressed when used in embedded product development. The characteristics of embedded product substantially differ from what agile was originally targeted for. Meeting real-time requirements of embedded systems is pointed out to be the most important difference that new agile methods should be able to support. In embedded systems, the role of architecture and up-front designing cannot be avoided. This requirement also leads to a need to find techniques that take into account the suitable amount of documentation and specification. Furthermore, it is pointed out by Ronkainen and Abrahamsson that top level documentation is important due to many different stakeholders involved in the project, and for this problem, it is pointed out that coordination and communication methods are required. The view that system-level documentation is required in embedded product development is also supported by [P3,P6,P12]. In [P12], a document-driven development approach is proposed where the importance of documentation is believed to be prominent especially in complex real-time systems. The main idea is in a coherent and appropriate documentation system that can be used to effectively share information between project stakeholders through a document repository.

Ronkainen and Abrahamsson [P16] also observe that experimenting cannot be avoided as the hardware constraints affect the code in an unpredictable way and that

the amount of the embedded software generated with experimenting is significant. In addition, it is pointed out that as the development progresses and more teams are getting involved (as the product gets more and more integrated), the embedded software has to become more rigid as all the changes made in the later stages of development ripple further and further to other teams and developers. This requires changing practices during the project, which is not supported at the moment. This discussion was on constrained embedded software development, but in this study, we also looked for ways to develop embedded products including hardware. This idea is suited well to the embedded product development where later stage changes have a huge impact on the project. One of the ways in reducing the impact is proposed by Punkka [24] where SW and HW co-design is emphasized. The point is in starting early with what you can and using, e.g., bread boards and evaluation boards and rapidly build a demo or a prototype of the product by experimenting. This should help the problem with later stage changes as the development needs to become more rigid only after demo and prototype rounds. At this point, most of the uncertainty and the whole product-related changes should have been done, and required changes would not ripple down too far.

Ronkainen and Abrahamsson also discuss the problems in test-driven approach. Testing is the cornerstone of embedded systems as most of the generated code needs to be tested against hardware and the code can be hardware dependent. It is pointed out that test-driven approach is problematic because the test environment is different in the embedded domain, e.g. it has more severe memory and performance constraints. A need for an appropriate test suite is pointed out, a view shared in [P4]. Work from [P23,P24] advances test-driven approach where the most suitable testing practices from XP are adapted into the embedded systems development. Smith et al. [P24] also point out that practices should take into account the constraints of embedded systems, e.g. refactoring should emphasize making improvements in speed and in lowering the power consumption rather than making the code modular or readable.

Another view was in an experience report from Waldmann [P28]. It was found that with many different customers requiring slight modifications to a product, the product should be customizable instead of developing a whole new product for all the different customers. A similar problem is addressed in [P2] where Cordeiro and Barreto discuss a customizable platform-based design method. The platform contains a customizable processor that can be modified and extended on different products (a platform is not suitable to be used in [P28], but the same need is observed in both). This method can be used when a similar product is modified slightly to different customers or with completely new products that share the same basic functionality. The key is to determine the functionality that is shared over projects.

We also studied what kind of evidences there are to support the suitability of using agile methods in embedded software or embedded systems development (RQ3). The results showed that most of the studied academic and non-academic articles were experience reports from the industry and expert opinions. We did not find any rigorous controlled experiments. Therefore, it seems that the evidences in the suitability of agile methods and pros and cons of the methods need more research.

Some articles discuss about new product development in large-scale organizations, where embedded systems development requires hardware and mechanics in addition to software. Kettunen and Laanti [P11] suggest that a large company should understand what it tries to achieve with agile methods, instead of focusing only on team and project level. One way to introduce agile methods to a large company is piloting in several teams [P7]. A good idea is to preserve some of the key practices, like requirements engineering practices, instead of starting from a scratch.

There were many experience reports which stated that agile methods could be used in the embedded domain. In most of these cases, the chosen agile method was adjusted to fit the business model by choosing agile practices that could be implemented easily and would support the development work. It seems that practices that were felt to bring the best return on investment to the development process were implemented first. Depending on the case, these prioritized practices vary a lot due to the different nature of the business cases and, possibly, due to different customer interface, technical maturity, or adopted working methods and competences of development teams. Generally, there were no opposing views that agile methods could not be used in the embedded domain. However, evidence about the suitability is mostly based on opinions and uncontrolled case studies with inadequate description of research methods that strict conclusion could be made. There was only one questionnaire on the usefulness of XP and Scrum in embedded development. It was found that to some extent agile practices were used by at least two thirds of the respondents and that those who were familiar with them found them useful. It is also noteworthy to point out that failures or bad experiences from adopting agile were not reported in any of the studies. All studies state that agile methods could be beneficial in embedded domain and that most experience reports are in favour of agile methods.

Our survey included not only the development of embedded software, but also embedded hardware and integrated circuits development, and we found out that very little has been done from the hardware development point of view. However, in the study concerning non-

academic papers, more discussions about agile methods for hardware development were found than in peer-reviewed academic articles. The studies close to hardware development typically concentrate on more abstract issues such as communication or requirement management. Therefore, there is a need for more rigorous research on utilization of agile methods in the actual development work of hardware and integrated circuit designers.

2 Conclusions

It was found that there are embedded domain-specific problems about agile methods that need to be solved before agile methods can be successfully applied to the embedded domain. To some extent, there are studies that address these issues, but the amount of evidence still remains scarce. Most of the studies address the issues of embedded software development. Some studies concerning embedded systems development were also found, but the amount of hardware-related agile studies remains low. Some discussions, however, were found among the non-academic articles concerning agile methods in embedded systems and hardware development.

Most of the found experience reports and case studies find no reason why agile methods could not - at least to some extent - be used for the embedded domain, but the lack of rigorous empirical research is a clear gap on the evidence of the actual benefits of agile methods in the embedded domain.

Selected academic articles

P1. C.O. Albuquerque, P.O. Antonino and E.Y. Nakagawa. An Investigation into Agile Methods in Embedded Systems Development. *Computational Science and Its Applications — ICCSA 2012*, Salvador de Bahia, Brazil, pages 576–591, June 2012.

P2. L. Cordeiro, R. Barreto and M. Oliveira. Towards a Semiformal Development Methodology for Embed ded Systems. *International Conference on Evaluation of Novel Approaches to Software Engineering*, Funchal, Madeira, Portugal, pages 5–12, May 2008.

P3. J. Drobka, D. Noftz and R. Raghu. Piloting XP on Four Mission-Critical Projects. *IEEE Software*, 23(6), pages 70–75, November–December 2004.

P4. M. Fletcher, W. Bereza, M. Karlesky and G. Williams. Evolving into Embedded Development. *Agile Conference*, Washington, DC, USA, pages 150–155, August 2007.

P5. B. Greene. Agile Methods Applied to Embedded Firmware Development. *Agile Development Conference*, Salt Lake City, Utah, USA, pages 71–77, June 2004.

P6. E. Gul, T. Sekerci, A.C. Yücetürk and Ü. Yildirim. Using XP in Telecommunication Software Development. *The Third International Conference on Software Engineering Advances*, Sliema, Malta, pages 258–263, October 2008.

P7. J. Heidenberg, M. Matinlassi, M. Pikkarainen, P. Hirkman and J. Partanen. Systematic Piloting of Agile Methods in the Large: Two Cases in Embedded Systems Development. *11th international conference on Product-Focused Software Process Improvement*, Limerick, Ireland, pages 47–61, 2010.

P8. P.M. Huang, A.G. Darrin and A.A. Knuth. Agile Hardware and Software System Engineering for Innovation. *IEEE Aerospace Conference*, Montana, USA, March 2012.

P9. P. Kettunen. Adopting key lessons from agile manufacturing to agile software product development – A comparative study. *Technovation*, 29(6–7), pages 408–422, July 2009.

P10. P. Kettunen and M. Laanti. How to Steer an Embedded Software Project: Tactics for Selecting the Software Process Model. *Information and Software Technology*, 47(9), pages 587–608, June 2005.

P11. P. Kettunen and M. Laanti. Combining Agile Software Projects and Large-scale Organizational Agility. *Software Process Improvement and Practice*, 13(2), pages 183–193, July 2007.

P12. Luqi, L. Zhang, V. Berzins and Y. Qiao. Documentation Driven Development for Complex Real-Time Systems. *IEEE Transactions on Software Engineering*, 30(12), pages 936–952, December 2004.

P13. P. Manhart and K. Schneider. Breaking the Ice for Agile Development of Embedded Software: an Industry Experience Report. *26th International Conference on Software Engineering*, Edinburgh, Scotland, pages 378–386, May 2004.

P14. C.E. Matthews. Agile Practices in Embedded Systems. *SPLASH'11 Workshops*, New York, USA, pages 249–250, October 2011.

P15. D. Morgan. Covert Agile: Development at the Speed of Government? *Agile Conference*, pages 79–83, Chicago, USA, August 2009.

P16. J. Ronkainen and P. Abrahamsson. Software Development Under Stringent Hardware Constraints: Do Agile Methods Have a Chance? *4th International Conference on Extreme Programming and Agile Processes in Software Engineering*, Genova, Italy, pages 73–79, May 2003.

P17. O. Salo, P. Abrahamsson. Agile Methods in European Embedded Software Development Organisations: a Survey on the Actual Use and Usefulness of Extreme Programming and Scrum. *IET Software*, 2(1), pages 58–64, February 2008.

P18. D. dos Santos Jr., I.N. da Silva, R. Modugno, H. Pazelli and A. Castellar. Software Development Using an Agile Approach for Satellite Camera Ground

Support Equipment. *Advances and Innovations in Systems, Computing Sciences and Software Engineering*, pages 71–76, 2007.

P19. J. Savolainen, J. Kuusela and A. Vilavaara. Transition to Agile Development – Rediscovery of Important Requirements Engineering Practices. *18th IEEE International Requirements Engineering Conference*, Sydney, Australia, pages 289–294, September–October 2010.

P20. J. Shalf, D. Quinlan and C. Janssen. Rethinking Hardware-Software Codesign for Exascale Systems. *IEEE Computer*, 44(11), pages 22–30, November 2011.

P21. A. Shatil, O. Hazzan and Y. Dubinsky. Agility in a Large-Scale System Engineering Project: A Case-Study of an Advanced Communication System Project. *IEEE International Conference on Software Science, Technology and Engineering*, Herzlia, Israel, pages 47–54, June 2010.

P22. M. Shen, W. Yang, G. Rong and D. Shao. Applying Agile Methods to Embedded Software Development: A Systematic Review. *2nd International Workshop on Software Engineering for Embedded Systems*, Zurich, Switzerland, pages 30–36, June 2012.

P23. M. Smith, A. Kwan, A. Martin and J. Miller. E-TDD – Embedded Test Driven Development a Tool for Hardware-software Co-design Projects. *6th International Conference on Extreme Programming and Agile Processes in Software Engineering*, Sheffield, UK, pages 145–153, June 2005.

P24. M. Smith, J. Miller and S. Daeninck. A Test-oriented Embedded System Production Methodology. *Journal of Signal Processing Systems*, 56(1), pages 69–89, July 2009.

P25. J. Srinivasan, R. Dobrin and K. Lundqvist. 'State of the Art' in Using Agile Methods for Embedded Systems Development. *33rd Annual IEEE International Computer Software and Applications Conference*, Seattle, Washington, USA, pages 522–527, July 2009.

P26. N. Van Schooenderwoert. Embedded Agile Project by the Numbers with Newbies. *Agile Conference*, Minneapolis, Minnesota, USA, July 2006.

P27. N. Van Schooenderwoert and R. Morsicato. Taming the Embedded Tiger – Agile Test Techniques for Embedded Software. *Agile Development Conference*, Salt Lake City, Utah, USA, pages 120–126, June 2004.

P28. B. Waldmann. There's Never Enough Time – Doing Requirements Under Resource Constraints, and What Requirements Engineering Can Learn from Agile Development. *19th IEEE International Requirements Engineering Conference*, Trento, Italy, pages 301–305, August–September 2011.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
The research reported in this article has been conducted as a part of AgiES (Agile and Lean Product Development Methods for Embedded ICT Systems) project. The project is carried out in collaboration with Finnish Institute of Occupational Health and industry partners BA Group, FiSMA, Lindorff Finland, LM Ericsson, Neoxen Systems, Nextfour Group and Nordic ID. The project is mainly funded by Tekes - the Finnish Funding Agency for Technology and Innovation.

Author details
[1]Business and Innovation Development (BID), University of Turku, Turku 20520, Finland. [2]Oy LM Ericsson Ab, Jorvas 02420, Finland. [3]Turku School of Economics, Department of Management and Entrepreneurship, University of Turku, Turku 20014, Finland.

Received: 17 April 2013 Accepted: 29 October 2013
Published: 15 November 2013

References

1. W Royce, Managing the development of large software systems: concepts and techniques, in *Proceedings of the IEEE WESTCON*, (Los Angeles, USA, August 1970)
2. C Larman, *Agile & Iterative Development: A Manager's Guide* (Addison-Wesley, Boston, 2003)
3. T Dybå, T Dingsøyr, Empirical studies of agile software development: a systematic review. Inf. Softw. Technol. **50**(9–10), 833–859 (2008)
4. Agile Manifesto. http://www.agilemanifesto.org
5. K Beck, *Extreme Programming Explained: Embrace Change* (Addison Wesley Professional, Boston, 1999)
6. K Schwaber, SCRUM development process, in *Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA)*, (Austin, Texas, USA, October 1995), pp. 117–134
7. DJ Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business* (Blue Hole Press, Washington, 2010)
8. J Warsta, P Abrahamsson, Is open source software development essentially and agile method?, in *3rd Workshop on Open Source Software Engineering*, (Portland, Oregon, USA, May 2003)
9. P Abrahamsson, O Salo, J Ronkainen, J Warsta, Agile software development methods: review and analysis. VTT Technical Report (2002)
10. BA Kitchenham, S Charters, Guidelines for performing systematic literature reviews in software engineering. version 2.3. *EBSE Technical Report*, EBSE-2007-01, Keele University, Keele, Staffs, UK, (2007)
11. VR Basili, G Caldiera, HD Rombach, The goal question metric approach, in *Encyclopedia of Software Engineering* (Wiley, Hoboken, 2002)
12. N Johnson, Top 8 people and resources for agile hardware development. http://www.axcon.dk/blog/process/top-8-best-people-and-resources-for-agile-hardware-development.htm. Accessed 16 January 2013
13. M Levison, Agile for hardware and embedded systems. http://agilepainrelief.com/notesfromatooluser/2008/12/agile-for-hardware-and-embedded-systems.html. Accessed 16 January 2013
14. T Punkka, Taming the big animal – agile intervention in a large organization. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.3719&rep=rep1&type=pdf. Accessed 16 January 2013
15. M Bartley, Agile development in hardware. http://testandverification.com/articles/agile-development-in-hardware/. Accessed 16 January 2013
16. J Grenning, Deep agile embedded panel questions – hardware. http://www.renaissancesoftware.net/blog/archives/42. Accessed 16 January 2013
17. N Johnson, Agile hardware development – nonsense or necessity? http://www.eetimes.com/design/eda-design/4229357/Agile-hardware-development-nonsense-or-necessity-. Accessed 16 January 2013
18. L Maccherone, Top 10 questions when using agile in hardware projects. http://maccherone.com/larry/2010/02/23/top-10-questions-when-using-agile-on-hardware-projects/. Accessed 16 January 2013
19. D Dahlby, Applying agile methods to embedded systems development. http://www.embuild.org/dahlby/agileEm/agileEm.pdf. Accessed 16 January 2013

20. Z Irani, Challenges of adopting agile in combined hardware and software environments. http://www.cprime.com/blog/2012/08/01/challenges-of-adopting-agile-in-combined-hardware-and-software-environments/. Accessed 16 January 2013
21. N Van Schooenderwoert, Embedded agile is different from vanilla agile. http://leanagilepartners.com/blog/blog/2011/10/10/embedded-agile-is-different-from-vanilla-agile/. Accessed 16 January 2013
22. MR Bakal, J Althouse, P Verma, Continuous integration in agile development. http://www.ibm.com/developerworks/rational/library/continuous-integration-agile-development/continuous-integration-agile-development-pdf.pdf. Accessed 16 January 2013
23. M Karlesky, G Williams, W Bereza, M Fletcher, Mocking the embedded world: test-driven development, continuous integration, and design patterns. http://www.methodsandtools.com/archive/archive.php?id=59. Accessed 16 January 2013
24. T Punkka, Agile hardware and co-design. http://www.ngware.eu/blog/papers/Agile2011_agile%20hardware%20and%20co-design_Punkka.pdf Accessed 16 January 2013

doi:10.1186/1687-3963-2013-15
Cite this article as: Kaisti *et al.*: **Agile methods for embedded systems development - a literature review and a mapping study.** *EURASIP Journal on Embedded Systems* 2013 **2013**:15.