2013 39th Euromicro Conference Series on Software Engineering and Advanced Applications

Kanban in software development: *A systematic literature review*

Muhammad Ovais Ahmad, Jouni Markkula and Markku Ovio
Department of Information Processing Science
University of Oulu
Finland
E-mail: {Muhammad.Ahmad, Jouni.Markkula, Markku.Oivo}@oulu.fi

***Abstract*— Using of Kanban in software development is an emerging topic. This systematic literature review was conducted in order to analyze the current trend of Kanban usage in software development and to identify the obtained benefits and involved challenges. The search strategy resulted in 492 papers, of which 19 were identified as primary studies relevant to our research. The main reported benefits of using the Kanban method were improved lead time to deliver software, improved quality of software, improved communication and coordination, increased consistency of delivery, and decreased customer reported defects. The reported challenges included lack of knowledge and specialized training as well as various organizational issues. Additionally, suggested practices were extracted from the primary studies and summarized for guiding the practitioners interested in adopting Kanban. The findings of this literature review are intended for helping researchers and practitioners to gain a better understanding of the current state of Kanban usage in software development.**

Keywords- kanban; software development; lean approach; systematic literature review

I. INTRODUCTION

In today's dynamic business and technology environment, organisations are continuously adapting their structures, strategies and policies in response to new demands. Operational costs are reduced and time-to-market is made shorter by increasing productivity. The increasingly important questions are, how to develop better and cheaper software and deliver it faster to fulfil continuously changing customer requirements. In existing literature, a number of approaches have been suggested for achieving these goals. Agile methods (AM) have been one of the solutions, which have gained popularity among the software industry. They have been considered as viable solutions for problems that appear to be hard to handle in traditional plan driven software engineering approaches. During the last decade, the Lean approach has had a major impact on the competitiveness of organizations through improvements in their process efficiency, and a reduction in their operational waste. Today, Lean is used in most global industries and virtually in all organizational sectors including the software industry. Lean software engineering approach has been derived from the lean manufacturing. Lean approach was developed and found to be successful in the car manufacturing industry [1]. Now it is among the fastest growing approaches used by the software development professionals. The key trait of lean is to eliminate all kinds of waste from the development. In order to eliminate and manage waste Poppendieck and Poppendieck [1] have proposed a set of principles, which are: build quality in, create knowledge, defer commitment, deliver fast, respect people and optimise the whole. In software development, all of those things that do not produce value for the customer are considered "to be waste". Furthermore, partially completed work, extra processes, extra features, task switching and defects are also considered to be waste [1].

One of the presented Lean tools is the Kanban approach to managing production operations [2]. In recent years, Kanban has become more popular in software development. Kanban approach is the most recent addition to the agile and Lean software development research area. A strong practitioner-driven movement supporting the idea of using Kanban in software engineering has emerged [3, 4]. Despite recently increasing interest in Kanban among practitioners, existing scientific literature seems to address it quite infrequently in the context of software development. Only a few studies on Kanban usage, how it is carried out in practice and what are its effects in software development, have been published. This motivates us to investigate the status of Kanban in software development, in terms of its presence in existing literature. This study is the first step to explore the Kanban approach in software engineering through a systematic literature review. This systematic literature review encompasses the current knowledge, gained benefits, and faced challenges while using Kanban in software development, and also identifies the needs and opportunities for future research in the area. This review will be highly important for practitioners who want to stay up to date with the state of research. On the other hand, it can provide new research areas and opportunities to be explored.

The rest of the paper is structured as follows. Section II describes the background and motivation of the study. Sections III discusses the methods used in this study. Section IV reports the findings of the review, along with the implications of the research. Section V provides a conclusion, with recommendations for further research.

978-0-7695-5091-6/13 $26.00 © 2013 IEEE
DOI 10.1109/SEAA.2013.28

II. KANBAN IN SOFTWARE ENGINEERING

Lean and Kanban approaches were introduced in the Japanese manufacturing industry in the 1950s. Kanban is a Japanese word meaning a signboard, and it is used in manufacturing as a scheduling system. It is a flow control mechanism for pull-driven Just-In-Time production, in which the upstream processing activities are triggered by the downstream process demand signals [2, S9]. It was successfully used in practice by Toyota. The basic idea behind Kanban usage is to execute the Lean thinking in practice; however Lean is more than Kanban [5, 6, 7].

The Kanban method in software development was originated in 2004, when David J. Anderson [9] was assisting a small IT team at Microsoft that was operating poorly. The Kanban method in software development drives project teams to visualise the workflow, limit work in progress (WIP) at each workflow stage, and measure cycle time [8]. The Kanban board provides visibility to the software process, because it shows assigned work of each developer, clearly communicates priorities and highlights bottlenecks. Additionally, its goal is to minimize WIP, i.e. develop only those items which are requested. This produces constant flow of released work items to the customers, as the developers focus only on those few items at given time. Kanban method aims to quickly adapt the process by using shorter feedbacks loops. The key impetus for the usage of Kanban's is focus on flow and the absence of obligatory iterations.

The current understanding of Lean software development is largely driven by practitioners' books [1, 9]. Maintaining the core aim of Lean approach, various Lean principles for software development have been presented and the situation is constantly evolving. Table I shows the Lean software development principles [1] and Kanban principles [9], which are known to the agile community. The Lean and Kanban principles appear to be largely overlapping, which reflects the same grounding.

TABLE I. LEAN AND KANBAN PRINCIPLES

Lean software development principles [1]	***Kanban Principles [9]***
Eliminate waste Build quality in Create knowledge Defer commitment Deliver fast Respect people Optimise the whole	Visualise the workflow Limit Work In Progress Measure and manage flow Make process policies explicit Improve collaboratively (using models and the scientific method)

The results of applying Kanban method in software development are expected to be highly positive, corresponding to the achieved advantages in the manufacturing industry. Due to this expectation, the adoption of Kanban method in the software development has gained strong practitioner-driven support. This expectation is based on Kanban's characteristics of adaptability (welcomes changes in requirements) and visualization (eases management by visualising the progress), as well as the prerequisites for its successful use (driving team members to cooperate and communicate) [8].

The application of Kanban method in software processes is expected to yield certain positive results, which are explained by Anderson [9]. Kanban limits work in progress according to team capacity, which balances demand against the throughput of team delivered work. It helps to visualise process problems, decrease defects and maintain a steady flow. By limiting work in progress a sustainable pace of development is achieved, yielding higher quality products and greater team performance. The combination of improved flow and higher quality software helps to shorten lead time, leading to regular releases that help in building trust among the customers.

In the past few years, a growing number of diverse teams and organizations using Kanban method have reported in various blogs and various discussion boards on the Internet. The proliferation boomed even more after first books on Kanban method in software engineering were published [9, 10]. One of these books is David J. Anderson's "Kanban" [9], where he introduced the concept of Kanban in systems and software development. Another popular book is "Scrumban" written by Corey Ladas [10], where he discusses the fusion of scrum and Kanban. It suggests that eligibility to use the Kanban approach in software development depends on the satisfaction of the following two axioms.

- It is possible to divide work into small value-adding increments that can be independently scheduled.
- It is possible to develop value-adding increments in a continuous flow, from requirement to deployment.

As we have mentioned above, the growing interest in Kanban in software engineering has been demonstrated by the recent publication of several books on this area. Successful histories of the manufacturing industry has convinced the software engineering professionals and researchers to adopt this approach. The next section presents the systematic literature review method adopted for this research and highlights the extent of Kanban in software development undertaken during the past decade journals and conferences.

III. RESEARCH METHOD

We designed this systematic literature review by following the guidelines given by Kitchenham [11]. According to these guidelines, review processes consist of several phases including the development of review protocol, identification and selection of primary studies, data extraction and synthesis and reporting of results. For this study, the review protocol was developed jointly by the authors of this paper while the corresponding author carried out the identification and selection of the primary studies following the specified protocol. All of the steps of the protocol are described below in this section. The objective of the review was to answer the following research questions:

- RQ1. What has been reported about Kanban in software development in the existing scientific literature?
- RQ2. What are the obtained benefits and challenges involved in Kanban usage in software development?
- RQ3. What are the suggested practices in existing Kanban studies?

A. Data sources and search strategies

The literature sources selected for the review included the following key electronic databases in the field:

- ACM Digital Library
- ABI/Inform (ProQuest)
- Science Direct (Elsevier)
- Springer Link (LNCS)
- Web of Science (ISI)
- IEEE Xplore - IEEE/IEE Electronic Library

Relevant studies were searched in the databases by using the following search strings:

1. Kanban AND software engineering
2. Kanban AND software development
3. Kanban AND challenges
4. Kanban AND software industry
5. Kanban AND software design

All of these search strings were combined by using the Boolean "OR" operator, which meant that an paper only had to include one of the strings to be retrieved. The structure of the search queries was the following:

1 OR 2 OR 3 OR 4 OR 5

B. Selection process

According to protocol, the papers were defined as eligible for inclusion in the review if they focused on Kanban usage in software engineering or software development. The inclusion criteria stipulated that the papers had to be published between the years 2000-2011, written in English, peer-reviewed and electronically available. Studies were excluded if their focus, or main focus, was not Kanban in software development. Additionally, ''lessons learned" papers (papers without a research question) and papers merely based on expert opinions were also excluded.

In assessing the quality of studies, we developed a checklist outlining the major quality criteria expected from the primary studies. We built the list based on quality criteria adapted from Kitchenhem et al. [11]. Fig 1 presents the systematic review process, which was carried out by the corresponding author of this paper to identify the primary studies, as well as the number of papers identified at each stage.

Initially, Step 0 of the search resulted in total of 1,828 papers. In Step 1, the titles of the obtained studies were read, in order to examine their relevance to our review. The papers with titles clearly showing that they were outside the focus of this study were excluded in this stage. For example, our search strategy included the term 'Kanban', which resulted in several hits on papers about Kanban in the manufacturing industry. Those papers were excluded, as the manufacturing industry is outside the focus of our study. Nevertheless, excluding decisions were not in all cases based only on titles, as some of the titles did not reveal the contents of the papers clearly enough. In these cases the papers were included for review in the next step.

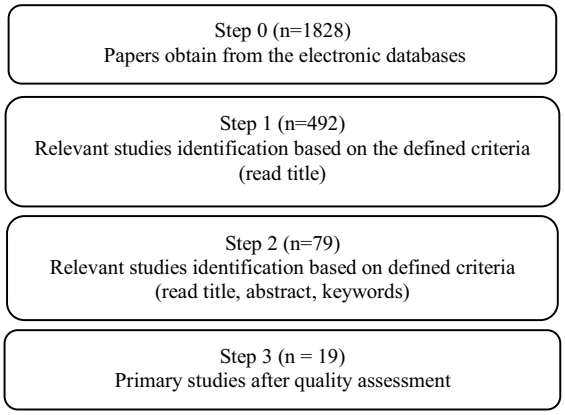

Fig. 1. Steps of the study selection process

In Step 2, the titles, abstracts and keywords of the remaining 492 papers were examined with regard to their relevance to our literature review. All of the papers not focusing on the Kanban method of software development were excluded in this step. We found that some of the abstracts appeared to be misleading, or revealed very little about the contents of the full papers. In such cases, the papers were included in the next step. This resulted in a total of 79 papers.

In Step 3, each of the remaining papers was assessed with regard to its quality and relevance to our study. The evaluation was done based on the following criteria: objective of the study, context description, research design, data collection and analysis, justification of findings, result applicability and use of references. Of the evaluated 79 studies 19 were finally accepted and included as the primary studies for our research. The rest of the papers were excluded because they did not pass the minimum quality threshold.

IV. RESULTS

In this section we present the analysis and outcomes of our systematic literature review. Our intention is to elaborate on the extracted data and to answer the research questions described in section III.

A. RQ1:What has been reported about Kanban in software development in the existing scientific literature?

We identified 19 primary studies on Kanban method to software development (see Appendix A). From the year 2000 to 2007, no studies dealing specifically with Kanban in software development were identified. Even though the search brought up some studies published during that time about using Kanban, they were related to integrated circuit

manufacturing systems and related manufacturing areas. The number of papers on Kanban in the software development context appears to have increased over the past few years, which can be seen from Table II.

TABLE II. PRIMARY PAPERS PUBLISHED ANNUALLY

Year	2000-07	2008	2009	2010	2011	Total
Papers	-	1	2	6	10	19
Percentage	-	5%	10%	32%	53%	100%

The major portion (53%) of Kanban studies on software development were published in year 2011. This trend can be considered an indicator of practitioners' and researchers' growing interest in using Kanban method in software development and reporting about it.

Furthermore, a high percentage of primary studies were published in conference papers, most of which are experienced based and qualitative studies. Table III shows the distribution of primary studies according to publication channel.

TABLE III. PAPER DISTRIBUTION ACCORDING TO PUBLICATION CHANNEL AND OCCURRENCE

Source	Type	No. of papers	Percentage
XP 20XX	Conference	6	32%
Agile 200X	Conference	6	32%
LESS 20XX	Conference	3	16%
IEEE TEM 20XX	Journal	1	5%
IEEE ICECCS 20XX	Conference	1	5%
Euromicro SEAA 20XX	Conference	1	5%
ICSSP 20XX	Conference	1	5%
Total		19	100%

The number of publications also varies according to study type. Of the studies, 47% were classified as experience reports, 10% were simulation studies, 32% were qualitative studies, and the remaining 11% were quantitative studies. Based on this, we can say that the major portion of published research consists of experience studies reported in recent conferences.

With respect to context, 84% of the studies were related to co-located projects, and in the remaining 16%, the contexts were not clearly defined in the papers. This indicates that the majority of the primary studies were conducted in co-located team settings. Additionally no study reports Kanban usage in distributed settings.

The studies include also a few simulation studies [S1] [S2], in which the strengths of Kanban have been compared with the waterfall model and Scrum. The results show that the Kanban workflow was managed through an effective control mechanism to limit the work in progress and minimize the lead time. The advantage of this approach is that the work is better controlled, so that the effects of errors are kept limited [S1]. On the other hand, in the waterfall model often projects fail to complete due to the difficulty in correcting errors, including errors in requirement [S1]. By using the Kanban method, simultaneous work in progress is reduced, which leads to easier balancing of the overall production flow. Additionally, Kanban is a pull system for visualising work, improving flow, reducing waste, and maximising customer value. It uses the rate of demand to control the rate of production, and passes demand from the end customer up through the chain of customer store processes. The simplicity of Kanban motivates the people and controls the project activities more efficiently [S6].

B. RQ2. What are the obtained benefits and challenges involved in Kanban usage in software development?

Generally, primary studies report a great number of benefits of Kanban usage in software development, which correspond the benefits presented by Anderson [9]. The core principles that are mostly used in primary studies are the visualising of work and limiting of work in progress, which help in the prioritisation of tasks and management of flow. Table IV shows the benefits achieved by using the Kanban method, as reported in primary studies.

TABLE IV. KANBAN BENEFITS

Overall benefits of Kanban	Primary studies
Better understanding of whole processes	S3, S4, S6, S12, S14, S15, S7
Improved software quality	S3, S11, S13, S14, S16, S17
Improved meeting of customer needs and customer satisfaction	S3, S4,S6, S10,S13, S19
Increased motivation of engineers	S3,S6,S7,S12, S17, S19
Improved communication/ coordination between stakeholders/ in team	S6,S12,S13,S14, S18
Bugs were fixed more quickly, WIP made it easier to handle blocking in work	S3, S6, S8, S12,S15
Increased software productivity	S3,S10,S12,S17
Problem solving (easy detection and removal of bugs)	S6, S10,S18
Reduced batch size	S3, S10,S12
Decreased time to delivery	S10,S16
Increased release frequency	S3,S10
Efficiently controlled software projects	S6,S12
Changes to requirements made welcome	S3,S6
Early feedback on features, without delays	S6
No massive documents (limited to customer request)	
Task approval from management not needed – approval gotten from customer in demos	

The main benefits of using Kanban are a better understanding of the whole process, improved quality and

better coverage of customer satisfaction. Due to its granular nature and the visualisation principle, it helps in understanding the whole work of the team and reduces batch size. One successful example is that of BBC Worldwide [S10], where the lead time to deliver software was improved by 37%, the consistency of delivery rose by 47%, and defects reported by customers fell 24% compared to the previous adopted agile method. As a result the overall performance of the development team was improved.

The simplicity of Kanban motivates teams to learn and, due to its lightweight process, increases team control over activities [S6]. Additionally, team communication and coordination with stakeholders is improved [S3, S6, S7] [S12]. The other team motivation factor is that more and more single tasks are decided on by low-level teams. The team itself creates approval and moves tasks in columns. The developers can approve each other's tasks, which is a sign of a self-organised and motivated team.

The visualisation of tasks with Kanban makes it clearer for developers to understand the overall direction of work, and helps them to manage the flow. This collective understanding of work motivates team members, which leads to more efficient work. It makes it easier to control bugs and see blocked tasks. When a task is blocking the flow of work, team members help each other to complete the task and move forward. This helps to improve team communication and coordination [S6, S12, S13, S14, S18].

The 'limit work in progress' principle of the Kanban method minimises context switching and lead time. Each member of the team works on assigned tasks and focuses only on these, which allows him or her to control individual work better and makes it easier to handle or avoid blocking tasks. It makes delivery faster and improves quality of work, reduces batch size and, at the same time, increases release frequency.

In limiting work in progress, prioritisation practice guides which work item should be pulled next in order to optimise value. Primary studies [S3, S6, S14, S10, S13] show that the prioritisation of tasks makes customer more satisfied. Changed requests concerning requirements are welcome, which also helps in getting earlier feedback, resulting in the minimisation of the delay of important features. Additionally, low-level team members decide on task management and prioritisation, which saves time. Furthermore, this helps to take decisions on small tasks to the team level, instead of waiting for top-level management approval. As Kanban does not involve any fixed plans, it helps in the avoidance of requirement cramming.

All of the primary studies reported various benefits gained by using the Kanban method. The studies that reported the most successful transitions were [S7, S10, S13, S14, S16]. The most prominent studies were [S10] and [S13], in which the following of Kanban practices as defined by Anderson [9] was shown to lead to success.

The Kanban method is relatively new in the field of software development. New innovation usually brings benefits, but at the same time, the adoption of new approaches introduces new challenges. Table V shows the challenges faced in the usage of the Kanban method, as reported in the primary studies.

TABLE V. CHALLENGES IN USING KANBAN APPROACH IN SOFTWARE DEVELOPMENT

Challenges	Primary Studies
Kanban can't work alone but requires supporting practices	S6, S7, S10,S12, S13, S17, S18
Hard to change organisational culture and philosophy	S3,S7,S15,S16, S17
Lack of specialised skills and training	S3, S4, S7, S16, S18
Collaboration and communication issues	S4, S8, S18
Motivating staff to use new practices	S4,S17,S18
Integration of Kanban projects with existing environments and processes	S3, S4
Hard to manage WIP	S4, S17
Hard to find a balance between the enforcing and caring roles	S3,S7
Kanban does not eliminate all waste	S5,S9
Hard to convince the top level management for the usage of new Kanban method	S7
Hard to select tasks according to priority	S4
Knowledge sharing	
Misunderstanding of project goals	
Need for guidelines for understand the process as a whole	S6
Kanban may not fit well with existing corporate standards	S10

The primary studies reveal that a mixed approach is often used in the adoption of the Kanban method. Kanban cannot be used alone, and requires supporting practices (e.g. agile practices).

In many studies, such as [S4] and [S17], Kanban was used along with other agile practices, and this mixed approach was problematic for many teams. For example, some found it hard to manage work in progress and faced difficulties in the prioritisation of tasks.

Motivating staff members to use Kanban was also challenging, because of organisational culture and the stickiness of other software development methodologies, as in [S3, S4, S7, S15, S16, S17, S18]. The lack of specialised skills and training, and the misunderstanding of core principles were some of the reasons for not adopting Kanban methods, as in [S3, S4, S7, S16, S18]. A lack of knowledge and skills was found to make collaboration and communication between Kanban team members and with other related stakeholders difficult in [S4], [S6] and [S18]. Moreover, limiting WIP was found to be problematic in [S4] and [S17], and waste was created instead of eliminated in [S5] and [S9]. This means that the necessary knowledge, training and support should be given, in order to overcome resistance, compatibility issues and knowledge sharing

problems. Convincing top-level management to use the Kanban method and limitations in budgets for process improvement e.g. sufficient budgets were needed for practices that changed systems, such as budgets to support the testing environment) were other challenges. as reported in [S7].

In the primary studies addressed in this paper, a number of other limitations of Kanban were also reported. For example, in [S4], physical distances and human language differences were found to lead to waste in the process of software development. Other issues were related to team learning and lack of standards [S10]. Although the Kanban approach is widely used in other fields, in software engineering it is relatively new. Therefore there is a need for guidelines for understanding overall processes [S6].

C. RQ3. What are the suggested practices in existing Kanban studies?

The primary studies suggest that the Kanban method needs to be implemented gradually in software development. It incrementally changes the way how work is done in an organisation. Patience throughout the process, which is very challenging for organisations, is essential.

According to [S6], visualisation alone does not guarantee success; Kanban is a basic controlling tool that needs to be supported with additional practices. In all of the primary studies, we see that there is a mixture of different agile practices used with the Kanban method. Hybrid models, integrating conventional agile practise with Kanban, are suggested for use. Mixtures of agile practices and Kanban principles were used in the primary studies, which showed incremental improvements in processes. The Kanban method should be used as a 'plug-in' with existing methods, rather than a replacement. Based on existing literature the following good practices can be summarised:

- Protect teams from external tasks during the actions phase.
- Consider ways of limiting work in progress.
- Create a culture for collaboration on solving tasks and problems.
- Use a visual board to make low team orientation visible and improve the shared mental model.
- Encourage team members to provide feedback to each other.
- In order to create value, note that all non-value added work is not waste, and that some non-value added waste is necessary.
- Have senior managers constantly work on value creation for the organisation, allotting time to teach and solve technical problem with teams. In short, provide technical leadership.
- Provide a clear vision for the whole team.

V. CONCLUSION

The aim of this study was to analyse the current usage of Kanban in software engineering, along with its benefits and challenges. There appears to be a lack of reported scientific research addressing Kanban usage in software engineering, but this study summarises the current knowledge in existing literature through a systematic literature review. Its results provide valuable information for future empirical studies on Kanban in software engineering, in both academic and industrial settings.

The literature review revealed that Kanban usage has been reported mainly at an abstract level. No studies were found that could clearly and deeply demonstrate Kanban method, and how they should be used in software development. The main findings of this study were classified into following categories, as presented below:

- The key motivation factors for adopting Kanban were simplicity, focus on the flow of work and no obligatory iterations. The primary studies demonstrate a trend of usage, transition and acceptance, from traditional software development methodologies to the Kanban method.
- The achieved benefits of using Kanban included customer satisfaction, improved software quality and lead time delivery, earlier feedback and reduction in customer defect reporting, improved communication between stakeholders and increased developer motivation. However, in most of the studies, these benefits were reported at the general level.
- There are meaningful differences between agile and Kanban, many teams found that blending the two approaches can create significant value for their organization. Typically Kanban was mixed with other agile practices by organisations. However, many studies reported that it was difficult to use such hybrid approaches. Educating people about Kanban method and changing organisational culture were some of the challenges revealed by the literature review.
- Incremental transition is recommended, starting with the basic idea of Kanban. So far, there is no unified method for deploying the Kanban approach, but the current literature suggests an incremental and mixed approach, involving other agile methods and proper team mentoring. Daily meetings were found to play an important role in information flow. During daily stand-up meetings, disturbances should be avoided, while leaving in the middle of daily stand up meeting should not be permitted. Educating staff about new approaches through specialised training is essential.

This paper identified a greater number of studies on Kanban method in software development have been published in recent years. The recent publication trend (see Table II) indicates that Kanban is gaining momentum in the software engineering domain. A high percentage of the primary studies appear to be experience reports (47%); and most of the studies involved pilot or small-scale projects, or recent transitions to the Kanban approach. Based on this, there appears to be a demand for more extensive and rigorous scientific research on Kanban in software engineering.

The review shows that a range of research methods have been applied. We need to employ both flexible and fixed research designs if we are to gain a deeper understanding of Kanban method. In fixed designs, the design is specified early in the research process, whereas in flexible designs, it is allowed to evolve during the research [14]. Both qualitative and quantitative data may be used in both fixed and flexible designs [14]. Edmondson and McManus [12], explained that a research design needs to fit the current state of research domain, and should go through three stages as shown in Table VI.

TABLE VI. CATEGORIES OF METHODOLOGICAL FIT IN FIELD RESEARCH

State of prior theory and research	Nascent	Intermediate	Mature
Research questions	Open-ended inquiry about a phenomenon of interest	Proposed relationships between new and established constructs	Focused questions and/ or hypotheses relating existing constructs
Type of data collected	Qualitative, initially open-ended data that need to be interpreted for meaning	Hybrid (both qualitative and quantitative)	Quantitative data: focused measures where extent or amount is meaningful
Theoretical contribution	A suggestive theory, often an invitation for further work on the issue or set of issues opened up by the study	A provisional theory, often one that integrates previously separate bodies of work	A supported theory that may add specificity, new mechanisms or new boundaries to existing theories

With regard to the Kanban method in software development, it is currently nascent and helps to reveal themes and issues that need further exploration. The immature nature of Kanban in software engineering results in exploratory qualitative studies having more open ended inquiries. Furthermore, they are needed to be explored in order for the unknown issues to emerge. More interest in the area will arise from such unexpected findings.

Based on currently available literature, the depth of Kanban usage and its impact cannot be reliably assessed, but there is an indication of a positive tendency toward using the Kanban method. As current knowledge on Kanban is limited, qualitative and quantitative studies in different environments and projects would be valuable. They would reveal the real strengths of the Kanban method in software development. Detailed guidelines related to Kanban usage in software engineering are required, as is more rigorous research on the core principles of the Kanban method (i.e. visualisation of workflow, limiting work in progress, measuring and managing flow) and its relationship to different dimensions of organisation (i.e. organisational culture, team leadership, self-managing teams, mechanisms for team learning and team motivation).

Furthermore, replications of some of the primary studies in various environments would be needed for validating their results. This would also allow cross-comparison, and strengthen the generalizability of the findings. It would be valuable to conduct research in real software development environments, in collaboration with academic organizations and companies. The usage of the Kanban method in various industry settings or different software development types could also be an interesting area for future research.

ACKNOWLEDGMENT

This paper is based on the work carried out in the Cloud Software Program. Cloud Software is a project by Tivit plc., one of the Finnish strategic centres for Science, Technology and Innovation (SHOK), and is funded by the Finnish Funding Agency for Technology and Innovation (Tekes).

REFERENCES

[1] Poppendieck, M. and Poppendieck, T, “Lean software development: An agile toolkit”, Addison Wesley, Boston, Massachusetts, USA 2003.

[2] Liker.J, “The Toyota Way”, McGraw-Hill, USA, 2004.

[3] Hiranabe, K, “Kanban applied to software development: From agile to lean” 2008, Retrieved on 4 May 2012, via: http://www.infoq.com/articles/hiranabe-lean-agile-Kanban.

[4] Shalloway, A., Beaver, G., and Trott, J. R, “Lean-agile software development: Achieving enterprise agility”, Pearson Education, Inc. / Addison-Wesley, Boston, Massachusetts, USA. 2009.

[5] Becker, M.; Szczerbicka, H., "Modeling and optimization of kanban controlled manufacturing systems with GSPN including QN," Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference, pp. 570- 575, 1998.

[6] Chai, L., "e-based inter-enterprise supply chain Kanban for demand and order fulfilment management," Emerging Technologies and Factory Automation, 2008. IEEE International Conference, pp.33-35, 2008.

[7] Ikonen.M, and P. Abrahamsson, "Anticipating Success of a Business-Critical Software Project: A Comparative Case Study of Waterfall and Agile Approaches", in Proc. ICSOB, pp.187-192 . 2010.

[8] Kniberg, H, “Kanban vs Scrum – how to make the most of both”, Retrieved on 19 July 2012, via: www.crisp.se/file-uploads/Kanban-vs-Scrum.pdf.

[9] David. J, Anderson, “Kanban: Successful Evolutionary Change for Your Technology Business”. Sequim, WA: Blue Hole Press, 2010.

[10] Ladas Corey, “Scrumban: Essays on Kanban Systems for Lean software development”, Seattle, WA, USA: Modus Cooperandi Press, 2009.

[11] Kitchenham. B, S. Charters, “Guidelines for Performing Systematic Literature Reviews in Software Engineering” EBSE Technical Report, EBSE-2007-01, 2007.

[12] A.C. Edmondson, S.E. Mcmanus, “Methodological fit in management field research, Academy of Management Review,” The Academy of Management Review, Volume: 32 Issue: 4, pp.1155-1179, 2007.

[13] Anderson, D. J, “Business drivers for Kanban adoption”, In Lean Software & Systems Conference, Atlanta, Georgia, USA 2010.

[14] Vigdis By Kampenes, Bente Anda, and Tore Dybå. “Flexibility in research designs in empirical software engineering”. In Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering (EASE'08), British Computer Society, Swinton, UK, 49-57, 2008.

APPENDIX A. PRIMARY STUDIES

[S1] D. Anderson, G. Concas, M.I. Lunesu, and M. Marchesi, "Studying Lean-Kanban Approach Using Software Process Simulation", in Proc. XP, pp.12-26, 2011.

[S2] Cocco. L, K. Mannaro, G. Concas, and M. Marchesi, "Simulating Kanban and Scrum vs. Waterfall with System Dynamics", in Proc. XP, pp.117-131, 2011.

[S3] Senapathi. M, P. Middleton, and G. Evans, "Factors Affecting Effectiveness of Agile Usage - Insights from the BBC Worldwide Case Study" ;in Proc. XP, pp.132-145, 2011.

[S4] Stray.V.G, N.B. Moe, and T. Dingsøyr, "Challenges to Teamwork: A Multiple Case Study of Two Agile Teams", in Proc. XP, pp.146-161, 2011.

[S5] Kettunen.P, "A Tentative Framework for Lean Software Enterprise Research and Development", In Proc. LESS, pp.60-71, 2010.

[S6] Ikonen.M, E. Pirinen, F. Fagerholm, P. Kettunen, and P. Abrahamsson, "On the Impact of Kanban on Software Project Work: An Empirical Case Study Investigation", In Proc. ICECCS, pp.305-314, 2011.

[S7] Nikitina.N and M. Kajko-Mattsson, "Developer-driven big-bang process transition from Scrum to Kanban", in Proc. ICSSP, pp.159-168, 2011.

[S8] Ikonen.M, "Leadership in Kanban Software Development Projects: A Quasi-controlled Experiment", in Proc. LESS, pp.85-98, 2010.

[S9] Ikonen.M, P. Kettunen, N.V. Oza, and P. Abrahamsson, "Exploring the Sources of Waste in Kanban Software Development Projects", In Proc.EUROMICRO-SEAA, pp.376-381, 2010.

[S10] Middleton, P.; Joyce, D., "Lean Software Management: BBC Worldwide Case Study," Engineering Management, IEEE Transactions, vol.59, no.1, pp.20-32, 2010.

[S11] Wijewardena, T., "Do You Dare to Ask Your HR Manager to Practice KANBAN? The Experience Report of an Offshore Software Company in Sri Lanka Introducing Agile Practices into its Human Resource (HR) Department," Agile Conference (AGILE), pp.161-167, 2011.

[S12] Polk, R., "Agile and Kanban in Coordination," Agile Conference (AGILE), pp.263-268, 2011.

[S13] Greaves, K., "Taming the Customer Support Queue: A Kanban Experience Report," Agile Conference (AGILE), pp.154-160, 2011.

[S14] Rutherford.K, P. Shannon, C. Judson, and N. Kidd, "From Chaos to Kanban, via Scrum", in Proc. XP, pp.344-352, 2010.

[S15] Birkeland.J.O, "From a Timebox Tangle to a More Flexible Flow", in Proc. XP, 2010, pp.325-334, 2010.

[S16] Taipale.M, "Huitale - A Story of a Finnish Lean Startup", in Proc. LESS, 2010, pp.111-114, 2010.

[S17] Willeke, E.R., "The Inkubook Experience: A Tale of Five Processes," Agile Conference, pp.156-161, 2009.

[S18] Shinkle, C.M., "Applying the Dreyfus Model of Skill Acquisition to the Adoption of Kanban Systems at Software Engineering Professionals (SEP)," Agile Conference, pp.186-191, 2009.

[S19] Kinoshita, F., "Practices of an Agile Team," Agile Conference, pp.373-377, 2008.