A cross-disciplinary systematic literature review on Kanban

University of Oulu
Department of Information Processing Science
Master’s Thesis
Adeyemi Bolaji
Date: 27.01.2015
Abstract

The issue of changes in requirements in software development due to fast and ever changing business condition attracts various software development processes. These development processes includes Agile, Lean software development and the recent addition of Kanban. Kanban method uses visual display (Kanban-board) to enhance software development by displaying various phases of the development process. Using explicit principles, Kanban method pulls tasks across different software development stages. This method is increasingly important, due to its many advantages, such as, reducing lead time, improving software quality, improving team communication, and increasing employee motivation. However, the knowledge of Kanban in software development is still limited.

This study used a systematic literature review method, to analyze relevant literature within the field of industrial engineering; taking into consideration that Kanban originates from the field of industrial engineering. The studies reviewed were published within 1997 and 2013. The search strategy identified 1552 papers, of which 53 were identified as relevant. These studies were then grouped into five themes—description, principles, benefits, performance and types. Based on these themes, the outcomes were then analyzed using existing knowledge of Kanban as related to software development.

The focus of this thesis was to uncover knowledge that can be useful in the application of Kanban methods to software development processes. The findings show the Kanban practiced in software development is somewhat the original Kanban in industrial engineering. Nevertheless, the review reveals several variants of Kanban method, driven by various production process environments. The findings also revealed that, the application of Kanban can reduce waste that exists in the form of extra-features. One of the implications of the research is the need to consider impact of context in the application of Kanban in software development process.

Keyword(s): Agile, Lean, Kanban, Software development, Kanban system, Systematic literature review.

Supervisors: Muhammad Ovais Ahmad, Dr. Jouni Markkula.
I will like to express my profound gratitude to all the people who have been with me during my studies and also during my thesis process. This process has been challenging but with your encouragements and supports of various kinds, I am able to arrive here, I say thank you. Also, during this time, I received lot of suggestions and feedbacks from my thesis supervisors. Special appreciation goes, Dr. Jouni Markkula, and Muhammad Ovais Ahmad. I truly appreciate your support, feedbacks and time. Also, I wish to thank, Harri Kaikkonen from Industrial Engineering Department, University of Oulu, Finland for his advice at the beginning of my research.
# Contents

Abstract .................................................................................................................................................. 2  
Foreword ................................................................................................................................................ 3  
1. Introduction ........................................................................................................................................ 5  
   1.1 Purpose ......................................................................................................................................... 6  
   1.2 Motivation .................................................................................................................................... 7  
   1.3 Research questions ..................................................................................................................... 8  
   1.4 Interdisciplinary nature of thesis ............................................................................................... 9  
   1.5 Research methodology .............................................................................................................. 9  
2. Related studies .................................................................................................................................... 11  
   2.1 Agile method ................................................................................................................................ 11  
   2.2 Lean software development ...................................................................................................... 13  
   2.3 Kanban and software development .......................................................................................... 17  
3. Systematic literature review ........................................................................................................... 21  
   3.1 Planning the review .................................................................................................................... 22  
   3.2 Conducting the review ................................................................................................................ 23  
      3.2.1 Data sources and search strategy .......................................................................................... 23  
      3.2.2 Selection of primary studies ............................................................................................... 25  
      3.2.3 Study selection procedure .................................................................................................. 25  
      3.2.4 Study quality assessment ................................................................................................... 26  
      3.2.5 Data extraction ..................................................................................................................... 27  
      3.2.6 Data synthesis ....................................................................................................................... 27  
4. Results of SLR .................................................................................................................................... 28  
   4.1 Overview of studies ..................................................................................................................... 28  
   4.2 Research question 1: How is Kanban described in terms of aims, process, principles, benefits and suggested practices? .......................................................................................................................... 28  
       4.2.1 Description and aims of Kanban ......................................................................................... 28  
       4.2.2 Kanban principles ................................................................................................................ 30  
       4.2.3 The benefits of Kanban method ....................................................................................... 31  
       4.2.4 Performance of a Kanban system ..................................................................................... 32  
       4.2.5 Suggested practices ............................................................................................................ 33  
   4.3 Research question 2: What types of Kanban systems exist in industrial engineering literature? ...................................................................................................................................... 34  
5. Result analysis ....................................................................................................................................... 39  
   5.1.1 Description of Kanban system ............................................................................................... 39  
   5.1.2 The benefits of Kanban method ............................................................................................ 40  
   5.1.3 Kanban system performance ............................................................................................... 41  
   5.1.4 Variants of Kanban systems .................................................................................................. 42  
6. Discussion ........................................................................................................................................... 43  
7. Conclusion .......................................................................................................................................... 46  
   7.1 Validity threats and limitations of study ..................................................................................... 47  
References ............................................................................................................................................... 48  
Appendix A. Structure of the review protocol .................................................................................... 54  
Appendix B. Quality assessment checklist .......................................................................................... 55  
Appendix C. Data extraction form ....................................................................................................... 56  
Appendix D. Matrix table .................................................................................................................... 57  
Appendix E. References for the primary studies ................................................................................ 58
1. Introduction

The issues of changes in requirements constantly confronting software development process, as a result of the fast and ever changing business conditions have attracted various software development processes. Highsmith and Cockburn (2001) noted that changes in product requirements that are out of the control of software developers occur more frequently in software development life cycle. The viewpoint of the traditional methods, such as prototyping, spiral model and the waterfall model, is that changes are eliminated early through meticulous planning and exhaustive requirement elicitions. However, restricting changes in software development process with the unstable business conditions, means “unresponsive to business conditions” contributing to “business failure” (Highsmith & Cockburn, 2001, p.120).

Cockburn (2002) notes that “winning in business increasingly involves winning at the software-development” (p. xxii), implying that software development process that allows changes in all the development phases are encouraged. Consequently, several software development processes have emerged in this regard. For instance, Agile software development methods such as Adaptive software development (ASD), Scrum, Crystal methods, Lean software development (Beck, Beedle & Van, 2001). The most recent addition to Agile and Lean realm in software engineering is Kanban (Anderson, 2010; Ahmad, Markkula & Oivo, 2013; Ahmad, Markkula, Oivo & Kuvaja, 2014). The introduction of Kanban to software development process is an effort to have a more responsive process in handling unstable business conditions.

Interestingly, Agile, Lean and Kanban concepts in software development were encouraged by their application in the industrial engineering field. Agile manufacturing was proposed in 1991, to increase the competitiveness of American industries in the 21st century (Nagel, & Dove, 1991; Cho, Jung & Kim, 1996). Also, its goal is to enable flexibility and responsiveness in order to utilize urgent and momentary market opportunities (Gunasekaran, 2010). Lean manufacturing was derived from Toyota production system (TPS), which was developed in 1950s. It is also called Just in Time production (JIT) (Womack, Jones & Roos, 2008). The central goals of TPS include producing “only the necessary quantity products, at the necessary time, in necessary quantity” (Sugimori, Kusunoki & Uchikawa, 1977, p. 553).

Further, among many approaches used in TPS to achieve its aims is to pull demand from the customers. In TPS pulling system, the downstream processes pull the parts needed from the upstream processes, in other words, the upstream process produces parts only when its downstream process demands it. Kanban, which means cards, was developed as an integral part of TPS to back this pulling system. In practice, Kanban enabled the movement or production of parts in TPS pulling system. The aims in a Kanban system include limiting work-in-process (WIP) as well as providing control to inventory.

In Agile software development, particularly in Extreme programming (XP), it is a common practice to use visual devices during the development process. These devices
are also called information radiators. They are used in displaying project information for everyone to see, and they are usually kept on the hallway or a place that will attract attentions. The key information on a typical radiator includes; status of project and task breakdown, using task cards. (Larman & Vodde, 2009.) In relation with the manufacturing Kanban cards, these cards are called “Agile Kanban” that may refer to actual task or requirement cards intended for filling (Poppendieck & Poppendieck, 2010). In addition, these cards are posted on a Kanban board. A Kanban board may refer to a board containing Agile Kanban cards (Larman & Vodde, 2009).

Anderson (2010), point out that, the use of these cards alone, does not innately suggest Kanban systems. Unless, there is explicit limit to WIP and signals which pull new task through the system, then in his words “they are merely visual control systems” (p. 14). This conclusion echoes Poppendieck and Poppendieck (2010) claim that Kanban systems are originally developed to limit WIP to increase throughput. Consequently, a Kanban system in software development contains Agile Kanban cards on a Kanban board, with the aims of limiting WIP and pulling work items through the development process. In short, Kanban is a tool used to facilitate software development process (Kniberg, 2009).

In software development, Kanban system is increasingly important to be ignored in both research community and practice. The reason for this attention owe to its derived benefits when applied to software development processes. In addition to limit WIP and visualization of the workflow, Anderson (2010) reports that Kanban systems can increase productivity, reduce lead time, as well as enhance predictability. Ahmad et al. (2013) report similar benefits.

Despite the growing number of studies on Kanban in software development, the suggested Kanban practices have proven to be inadequate (Ahmad et al., 2013; Corona & Pani, 2013; Nikitina & Kajko-Mattsson, 2011; Wang, Conboy & Cawley, 2012). Hence, evidence of difficulties relating to both implementation and operational issues exist. Also, tackling these challenges require additional understanding of Kanban, since the current state of the art on the subject in software development literature is relatively limited (Wang et al., 2012).

1.1 Purpose

In improving software development process, Evidence-Based Software Engineering (EBSE) was recommended by Kitchenham, Dybå and Jørgensen (2004). The goal of the current study aligned with the aim of EBSE, which is:

“To provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software.” (Kitchenham et al., 2004, the goal of EBSE section, para.1)

Considering, the limited knowledge of Kanban in software development literature, this research study would explore Kanban related literature of industrial engineering. Kanban was originally from the industrial engineering field. So seeking more
knowledge that can advance the understanding of the Kanban system, may consequently, inform its decision making process in software development.

1.2 Motivation

The main reasons that motivated the current research study are: first, there is evidence in the existing literature on Kanban (Corona & Pani, 2013), suggesting that Kanban knowledge (in software development) is still limited. Secondly, conducting more research on Kanban with the aim of advancing its knowledge can be regarded as adherence to Lean principle of “continuous improvement” mentioned in (Poppendieck & Poppendieck, 2003). These motivations are discussed in more detail subsequent paragraphs.

Wang et al. (2012) responded to the limited knowledge of Lean application strategies, in the context of software development. They conducted secondary data analyses –mainly experience reports. The aim was to uncover the implementation strategies of Lean approaches in software development. The results reveal that, developers tailor Lean strategies to suit their particular context. However, they disclosed the lack of roadmap for the application of Lean approaches. Similarly, Ahmad et al. (2013) conducted a systematic literature to disclose, what have been reported about Kanban in terms of interest, challenges, as well as suggested practice. The findings uncorked not only Kanban approach’s benefits, but also the challenges software developers encounter in applying Kanban in software development. Amidst the identified challenges are; lack of specialized skills and training, hard to manage WIP, hard to select task according to priority and need for guidelines to understand the process as a whole. These challenges can effectively be mitigated if there is a roadmap to Kanban implementation in software development. In short, their finding echoes the recommendation of Wang et al. (2012) as to the need for detailed guidelines.

Further, Corona and Pani (2013) reviewed state-of-art of the adoption of Lean related methods. Their focus mainly was Lean-Kanban board-related issues. The data analyzed were mere three practitioners’ books and web sources. This implies that, the research community on the subject has relied on practitioners’ books, for decision making or updates. Although, practitioners’ books are valuable but can be subjective.

Within the framework of ‘Lean thinking’ continuous improvement is an important activity in achieving leanness. It is one of the Lean principles that made Toyota’s production concept a standard to emulate today. (Womack et al., 2008.) In addition, Poppendieck and Poppendieck (2003) emphasize the importance of continuous improvement in Lean software development. The effort in the current study is toward improving Kanban knowledge in software development, thus, reacting to the continuous improvement principle. Also learning can resort to continuous improvement, and Ikonen, Pirinen, Fagerholm, Kettunen and Abrahamsson (2011, p.313), believes there could be “some potential for more cross-disciplinary learning since Kanban has been used for decades in the manufacturing environment.”
1.3 Research questions

To adequately respond to the suggestions of both Wang et al. (2012) and Ahmad et al. (2013), additional knowledge outside software development domain can be valuable. Consequently, this study has chosen the industrial engineering field as an external source for additional understanding on the subject. This decision was informed by the fact that, Kanban originated from that field; hence, its maturity level is relatively higher (Ikonen et al., 2011). Specifically, the research question addressed in this study is:

*What knowledge of Kanban in industrial engineering literature can be used to improve its application in software development?*

The supporting research questions are;

- **Research question 1:** How is Kanban described in terms of aim, process, principles, benefits and suggested practices?
- **Research question 2:** What types of Kanban systems exist in industrial engineering literature?

The rationale in investigating research 1, is that, the need to seek more understanding of Kanban in software development is evident in these studies (Ahmad et al., 2013; Corona & Pani, 2013; Wang et al., 2012; Ikonen et al. 2011). While the second research, is based on the assumption that Kanban approach must have evolved to fit different production or operational processes in the Industrial engineering field. So, the anticipated knowledge may uncover production or operational process, similar to software development.

These supporting questions were formulated with the use of PICO (Population, Intervention, Comparison, and Outcome) criteria. PICO is used in the medical field to formulating research question. Researchers in the medical field, are encouraged to seek this effectiveness of a particular treatment from three different perspectives—Population (i.e. the people that received the treatment), Intervention (i.e. type of treatment), Comparison (i.e. intervention is compared to other similar interventions), and the Outcomes (i.e. outcomes of interventions). (Kitchenham, 2007.) However, this study used PICO as explained by Kitchenham (2007) to formulate the sub-research questions.

The adaptation of PICO criteria to suit the purpose of this study produced PIO (Population, Intervention, and Outcome). The intervention in this case is *Kanban*, and the population using this intervention is *industrial practitioners*. In this study, the Kanban’s intervention is not being *compared* with other interventions in industrial engineering literature. According to Kitchenham (2007), *outcomes* are factors of interest to practitioners. Therefore, the evidence of Kanban reported in terms of *process, aims, principles, benefits, implementation strategies and types of Kanban systems* are factors of importance in this study.
1.4 Interdisciplinary nature of thesis

The research approach employed in this thesis is recommended, when the solution to the research problem is outside the scope of a particular discipline (in this case software development domain) (Repko, 2008). Palmer (2001) reports that, “The real-world research problems that scientists address rarely arise within orderly disciplinary categories, and neither do their solution” (Palmer, 2001, p. vii). Conboy and Fitzgerald (2004) used similar approach with the aim of resolving the inconsistencies in the usage of ‘agile’ in the context of software development. They conducted a literature review across multiple disciplines: Manufacturing, Finance, Management, Labour and Marketing, etc. Their findings facilitated a robust conceptual framework for agility in software development context.

Although, the differences between software engineering and the industrial engineering disciplines are acknowledged, as noted by Huber and Morreale (2002) that “each discipline has its own intellectual history, agreements, and disputes about subject matter and methods” (p. 2). Repko (2008) adds more comprehensive remark that “each discipline has its own defining elements-phenomena, assumptions, epistemology, concepts, theories and methods- that distinguish it from other disciplines.” (p. 4). However, it is assumed that the disciplinary difference as regard the phenomena under study is considered narrowed. Since the phenomena in question was originally from the industrial engineering field as claimed in these studies (Anderson, 2010; Sugimor, Kusunoki, Cho & Uchikawa, 1997; Womack et al., 2007).

Further, cross boundary studies are a metaphor associated with interdisciplinary studies. Admittedly, till date, no unanimous definition of interdisciplinary studies within the research community. (Repko, 2008.) However, the underlying idea of this thesis aligned with this definition:

“Interdisciplinary studies is a process of answering a question, solving a problem, or addressing a topic that is too broad or complex to be dealt with adequately by a single discipline, and draws on the disciplines with the goal of integrating their insights to construct a more comprehensive understanding” (Repko, 2008, p. 16)

Addressing the research problem with the software engineering literature alone, will be ineffective and unprosperous at the moment due to limited knowledge of the phenomena under study, hence, complex in software development. The aim in this section is to inform the reader that, the approach used in this thesis (investigating in the industrial domain) is rooted within the body of knowledge.

1.5 Research methodology

For the purpose of addressing the research problem, the research strategy is in two phase. First, the industrial literature based on the topic will be reviewed, using Systematic literature review (SLR) method. SLR is the primary method used in synthesizing best available evidence (Kitchenham, Pretorius & Budgen, 2010; Petticrew & Roberts, 2008). This current study will employ the SLR guidelines proposed by Kitchenham (2004) and Kitchenham and Charters (2007) to accomplish this phase.
Secondly, to identify areas in which the results can inform, result analysis will be conducted between the results of the SLR and the existing Kanban knowledge in software development. The analysis should allow the additional contribution of this study.
2. Related studies

The current study is situated in the space of software development process, with focus on Agile, Lean and Kanban methodologies. Section 2.1 presents an overview of Agile software development methods, to show their rationale and types. Section 2.2 presents an overview of Lean software development. Lastly, section 2.3 presents an overview of Kanban and its application to software development. Kanban method to software development is the motivated the current study.

2.1 Agile method

Agile methods emerged as a result of the shortcomings of the traditional methods, such as the spiral model. The traditional methods assumed complete customers’ requirements right from the start of the project (Highsmith, 2002). However, customers may at the beginning of the project have a vague idea of what they wanted, as a result, unable to submit complete requirements at the start. Similarly, the idea that requirements are fixed is inaccurate, as customers’ requirements are constantly changing and limiting the customers right to choose, or their expressions at any stage of the project is inappropriate. Additionally, the extensive documentation in the traditional methods increased time to market (Beck et al., 2001). For these reasons, different methods have emerged, for example, the spiral model (Boehm, 1988) and the Rapid application development (Martin, 1991). However, these methods failed to thoroughly address the issues of uncertainties in software development, particularly, changes in customers’ requirements (Highsmith & Cockburn, 2001). On the other hand, Agile methods emerged to manage the “incompleteness of communication” (Cockburn & Highsmith 2002, p.1) acknowledging the uncertainties in the development process.

Among the several methods in this category are- Extreme Programming (XP), Crystal Clear, Dynamic System Development Methods (DSDM), Feature Driven Development (FDD), Scrum (Cockburn, 2002). These methods, in contrast to the heavyweight plan-driven processes, have shorter development cycle; promote humanized software development process, and reduce cost of development (Lu & DeClue, 2011). These methods were first categorized as Agile methodology by the Agile Alliance group in 2001 (Cockburn & Highsmith, 2002). The Alliance created the Agile Manifesto, which summarizes the common similarities and beliefs underpinning these methods. According to Cockburn and Highsmith (2002, p.217), the Agile Manifesto reads:
"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more."

Dynamic systems development methods (DSDM) were introduced primarily for producing what business need, and at the time it needs it. Also, the development process is implemented in order to reduce time to market. (Stapleton, 2003.)

The adaptive software development (ASD) method is regarded as a lightweight development method that allows unceasing changes in requirements. The development process adheres to a flexible or dynamic lifecycle (speculation, collaboration and learning) approach, contrast to the plan driven traditional methods. ASD was developed to cater for product development with “high speed, high change, and high uncertainty”. In any project, planning is essential to achieve the desired results, which is also enabled by certainty. However, the business climate is sometimes unpredictable and uncertain, so this issue is handled by speculating, which accepts ‘trial and error’ by delivering short delivery cycles and iteration. Speculation is an approximation of reality, but good enough to manage the project. Another key construct of the ASD model is collaboration, in an unstable environment, developing a sophisticated application may require high information flow, as well as knowledge sharing. ASD recommends collaboration as a means to mitigate the problem of information flow and knowledge sharing, thus, facilitates the delivery of working products. The last construct is continuous learning which is assumed that responding to feedbacks will in return avert subsequent mistakes, which will in effect amount to the improvement of the development process. (Highsmith, 2002.)

Scrum an example of adaptive development method empowers development team members. The team members have the power to decide on how they will accomplish the assigned task within each increment. (Schwaber, 2000.) Schwaber (2000) defines Scrum as “a method that aims to help teams to focus on their objectives. It tries to minimize the amount of work people have to spend tackling with less important concerns” p.35. Bureaucracy is substantially reduced by the direct feedback given during the regular meeting; this is consistent with Beck et al. (2001) remark about Agile methods.

With respect to the efficacy of Agile methods, Lindvall, Basili, Boehm, Costa, Dangle, Shull, Tesoriero, Williams, and Zelkowitz (2002), conducted an empirical study to investigate the effectiveness of Agile methods in terms of environment and projects. Their findings were gathered using a virtual eWorkshop. The total number of participants was eighteen and they were given the Agile manifesto as a background material. Their finding identifies three key success factors of Agile methods - culture, people, and communication.
The central factors in the definition of agility according to Beck et al. (2001) are: rapid and adaptive to change, communicating with all stakeholders, treating customers as team members, to empower the team to make own decisions. Conboy and Fitzgerald (2004) conducted a study with the aim of correcting the inconsistency usage of agility in software development. So, they instance several disciplines outside software engineering that apply the same concept, agility. As a result, Conboy and Fitzgerald (2004) define agility as:

"The continual readiness of an entity to rapidly or inherently, proactively or reactively, embrace change, through high quality, simplistic, economical components and relationships with its environment." p.39

Qumer and Henderson (2006) offer an extended definition of agility using these variables; speed, flexibility, leanness, learning and responsiveness. Further, they propose a tool to measure agility and to appraise Agile methods. The tool can enhance the evaluation of Agile methods under the lens of four perspectives: method scope characterization, agility characterization, Agile values characterization and software process characterization (Qumer & Henderson-Sellers, 2006). In addition, Kruchten (2001) also agreed that, the unique and robust feature of Agile methods is flexibility. The view of flexibility and leanness as the core principles of Agile methods is supported by the aforementioned definitions of agility.

2.2 Lean software development

Lean concept originated from Toyota car production system (TPS). The production system consists of two unique features. The first is Just-In-Time production which means producing “only the necessary products, at the necessary time, in necessary quantity”. The second is a system that promotes respect-for-human, this system uncovers workers’ full potentials via workers’ participation, in the decision-making process. The driving force of this idea was the need to reduce the cost of production incurred by waste. (Sugimori et al., 1977.) Womack and Jones (1990) coined Toyota production system as Lean production (Anderson, 2010).

Further, researchers interested in Lean production have succeeded in extracting and defining the key principles of this technique as practiced by Toyota Company. Liker (1997) described the fourteen principles of Toyota Production System, using four concepts- Philosophy, Process, People and Partners, solving problems from the cause. Womack and Jones (1996), report that applying Lean technique will improve production to a considerable level. Typically, companies that employ this system, continuously seeking reduction in cost of production promotes zero defects and zero inventories. Womack et al. (2007) present the principles underpinning Toyota Production System (TPS), and the application of the Lean technique.

Poppendieck and Poppendieck (2003) were the first to adapt these Lean principles to software development. In addition, they propose twenty-two set of tools which were compared to Agile principles. The popularity of Lean software development among the research community and practitioners is attributed to them. Poppendieck and Poppendieck (2003) contribute immensely to the study of Lean software development.
(LSD). However, Lean has been recommended by earlier studies to software development (Hamilton 1999; Middleton, 2001; Raman, 1998).

The work of Poppendieck and Poppendieck (2003) was partly influenced by industrial engineering literature. Their claim that Lean technique can significantly improve productivity, reduce lead time, and reduce defects is consistent with the remarks in (Hamilton 1999; Middleton, 2001; Raman, 1998). Poppendieck and Poppendieck (2003) developed seven principles which are seen as the thrust of Lean software development (LSD), and these principles are listed below:

1. Eliminate waste
2. Build in quality
3. Create knowledge
4. Defer commitment
5. Deliver fast
6. Respect people
7. Optimize the whole

The ultimate goal of Lean is to identify waste and then make corrective action to eliminate it, while providing customer satisfaction. Poppendieck and Poppendieck (2003), contrast the sources of waste in the manufacturing and software development, as shown in Table 1.

**Table 1. Sources of waste (Poppendieck and Poppendieck, 2003).**

<table>
<thead>
<tr>
<th>Sources of waste in manufacturing</th>
<th>Sources of waste in software development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overproduction</td>
<td>Extra features</td>
</tr>
<tr>
<td>Inventory</td>
<td>Requirements</td>
</tr>
<tr>
<td>Extra processing steps</td>
<td>Extra steps</td>
</tr>
<tr>
<td>Motion</td>
<td>Finding information</td>
</tr>
<tr>
<td>Defects</td>
<td>Defects not caught by test</td>
</tr>
</tbody>
</table>

The central idea of Lean is to promote steady production, while reducing waste. In addition, only value adding activities are allowed under Lean, and non-value adding is considered waste (except necessary). It is believed that, eliminating non-value activities (waste) will promote value adding activities. In short, the overall goal is to reduce cost of production, while satisfying customers with quality products. (Sugimori et al., 1977; Womack et al., 2008.) The assumption that applying Lean techniques brings several advantages could be the motivation behind its application in software development process.

Dybå and Dingsøyr (2008) conducted an empirical study to uncover current knowledge on Agile development with respect to benefits, limitations and strength of evidence that supports the findings. The study classified LSD as an instance of Agile method, other studies accepted this argument (Karvonen, Rodriguez, Kuvaja, Mikkonen & Oivo, 2012; Staats, Brunner & Upton, 2011).
A large and growing body of literature has studied the applicability of Lean to software development process. For example, Middleton (2001) conducted a participatory action research, in an organization. Two teams (A and B) were formed and each consisted of an analyst and a programmer. Both teams were working on same project-financial and management information system. Although, before using Lean techniques, both teams developed standard operating procedure. The study shows that applying Lean techniques to software development can improve quality and motivate workers. The study however, finds no evidence suggesting that Lean techniques are inapplicable to software development. (Middleton, 2001.)

Middleton and Joyce (2012) investigated applying Lean to software project management. The focus was to determine how Lean concepts can be applied to software project management. Using a case study of an actual implementation process, they analyzed both quantitative and qualitative data. The result supports the proposed study hypothesis— that Lean will improve productivity and the whole development process. For example, the manner in which tasks were done improved considerably, and also the rate of task influx into the development process was significantly reduced. As a result, the company was able to deliver a quality product incrementally to their customers.

Staats et al. (2011) also investigated the application of Lean to software development. The core idea underpinning Lean is the ability to continuously identify problems and attempt to solve them. They made two recommendations that could improve problem solving approaches in fields undertaking different approach than the automobile industry. These propositions are - (1) “Problems should be identified as frequently as possible, as early as possible.” (2) “Problems and solutions should be kept together in time, space, and person.” (P.388). In addition, they argue that Lean is applicable to knowledge work. Their finding in respect to the applicability of Lean to software development is supported by Raman (1998). However, they also report that the high level of uncertainties and the absence of repetitive work, in the knowledge work are considerable challenges in applying Lean in those fields.

According to Nightingale and Mize (2002), adopting Lean requires process assessments. They present a framework that could facilitate the adoption of Lean in software companies. The framework was developed in MIT by Lean advancement initiative; called Enterprise Self-Assessment Tool (LEAST). The authors consider LEAST appropriate gauge for companies interested in adopting Lean. To adapt this framework to software, the authors reviewed industrial literature coupled with LEAST framework. The findings highlighted management involvement as an important factor for a successful Lean implementation. They hold the view that LEAST can enhance software industry in adopting Lean. In addition, to determine how to apply Lean techniques to software development, Jonsson, Larsson and Punnekkat (2013) conducted a systematic literature review. They identified nine influential sources usually referred to when discussing Lean in software development context. The study synthesizes forty four distinct Lean principles. Further, they categorized the key concepts discussed in the literature as- main philosophy (e.g. long-term decision, continuous improvement), process (e.g. value, value stream, flow, pull), people (e.g. mentorship leadership) and technology (visualization, powerful tools).

Still in the direction of Lean application to software development, Wang et al. (2012) investigated the ways in which LSD principles are been applied in Agile software
development. The study was a response to the limited knowledge of Lean application in software development. Wang et al. (2010) argue that Lean software development consists of Lean concepts, Lean principles and Lean practices. Lean concepts are referred to as the mindset of providing valuable products through appropriate value-stream, for instance value, value stream, flow, pull and perfection. Lean principles on the other hand are the tools used in identifying and eliminating waste. Lean practices are considered as the activities that help implement Lean principles, for example, addressing bottlenecks, using Kanban board to visualize workflow. Furthermore, to uncover the ways used for Lean application, Wang et al. (2012) conducted secondary data analysis of thirty experience reports published in Agile software development conferences. The result identifies six different ways shown in Table 2.

Table 2. Lean software application strategies in Agile software development (Wang et al. 2012).

<table>
<thead>
<tr>
<th>Type code</th>
<th>Lean Application Types</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Non-purposeful combination of Agile and Lean</td>
<td>The hybrid of Lean and Agile, with no particular difference between the two</td>
</tr>
<tr>
<td>B</td>
<td>Agile within, Lean out-reach</td>
<td>Using Lean to communicate outside an Agile setting</td>
</tr>
<tr>
<td>C</td>
<td>Lean facilitating Agile adoption</td>
<td>Using Lean to support Agile method</td>
</tr>
<tr>
<td>D</td>
<td>Lean within Agile</td>
<td>Using Lean principles to enhance Agile processes</td>
</tr>
<tr>
<td>E</td>
<td>From Agile to Lean</td>
<td>Using Lean to extend Agile methods.</td>
</tr>
<tr>
<td>F</td>
<td>Synchronising Agile Lean</td>
<td>Using Lean principle (e.g. Kanban) with Agile methods, in a synchronized fashion.</td>
</tr>
</tbody>
</table>

In Table 2 the Lean application strategies and their corresponding definition is illustrated. Wang et al. (2012) inform that, despite these unassociated categories, an organization can use each of these strategies based on their organizational needs. They advised that development context, project objectives, constraints of the project and knowledge on both Agile and Lean practices should be considered before opting for ‘leagile’ software development.

Rodriguez et al. (2013) investigated the interpretation and implementation of Lean principle in software development companies. Their result highlights the Lean principles that are achievable in software development as: “creating a culture of continuous improvement, involving people in the transformation, creating a team culture” (p.105). The study, however, identifies three principles of Lean that are still difficult to achieve in software development context as: achieving flow, transparency, and introducing the practice of continuous learning. In addition, the underlying idea behind Lean thinking is to eliminate waste by engaging in activities that will add value to the customers’ products. Thus, in the context of software development, they, argue that, introduction of product owners, upstream to downstream production and transparency are crucial in creating value-oriented activities.

Similarly, Pernstål, Feldt and Gorscheck (2013), identify and analyze status quo of Lean product development within the software engineering literature. The finding uncovers that, many studies address waste elimination and establishment of flow into software development process. Surprisingly, about 1.8 percent of the relevant studies reported the
combination of Lean and Agile principle and none was found on the feasibility or hybrid of Lean and the traditional methods. Also, the study reports the lack of other Lean principles and practices, for example, “the role of product manager, supplier integration, building the culture of continuous improvement, or tools for standardization and organizational learning” (Pernstål et al., 2013, p.2814). The study confirms the need for more research on Lean principles in software development context, particularly, in large-scale software development. In addition, they echo the inconsistencies in terminology, lamenting that one of the reasons they found only thirty eight papers relevant was because different terms are used for the same concept.

2.3 Kanban and software development

Kanban is increasingly becoming the central Lean practice evident in software development (Anderson, Concas, Lunesu & Marchesi, 2011). Kanban was introduced to software development by Anderson (2010). Who wanted to protect his team from influx of tasks from the management (causing task switching), to achieve continuous development pace and also to scale up Agile with little or no resistance to change. Applying Kanban to his software development process helped in achieving all these objectives. Kanban, has subsequently, attracted many attentions from both researchers and practitioners in the field.

Kanban in Japanese means “visual card” or “signboard.” The idea of Kanban is to support JIT production system to achieve its objectives, by implementing its pulling policy. For JIT the goal is to decrease WIP inventory, decrease costs by eliminating waste, and increase quality, while the primary goal of Kanban is to control work-in-process inventory. The idea is that, controlling inventory will expose both the areas that need improvements and waste in the production process. Hence, if corrective measures are applied, it results into JIT achieving its objectives. (Sugimori et al., 1977.) The principles of Kanban include: visualize workflow, limiting WIP, making explicative policies, recognize process improvement opportunities (Anderson, 2010).

In software development, Anderson et al (2011) define “Kanban software process as a WIP limited pull system visualized by the Kanban board” (p. 14). In addition, Kanban is the method used in incremental software development process. The goal of Kanban is to implement visual display of software development process to support Lean thinking of see the whole. Also supporting the rationale of the information radiators (Larman & Vodde, 2009) in Agile software development. It also provides restraints to control WIP and reveals bottlenecks. Further, Anderson (2010) identifies five important principles for implementing a successful Kanban system in software development process:

- Visualize the workflow
- Limit WIP by reducing the numbers of cards in each columns
- Measure and manage flow
- Explicit process policies
- Improve continuously

The figure 1 illustrates software development processes using the Kanban-board. The software development processes are illustrated using a column on the Kanban board.
The board is divided into columns, such that each column represents a separate phase in the development process.

![Kanban board diagram](image)

**Figure 1.** Example of a Kanban board (Brodzinski, 2009).

The board depicted in Figure 1 has five development phases and each one is able to receive inputs—constituents of the project, from their respective preceding phases. The total numbers of WIP limit in each stage are defined by integers. The rule is that, the process cannot take in new work item if it has reached its WIP limit. The first column in Figure 1 is the backlog, which contains all the tasks that would pass through the development process. The Kanban board is the essential component supporting its transparency and its power of waste reduction. It provides transparency of the project to a stakeholder, reveals queues and importantly, waste in the system. Consequently, driving team’s continuous improvement process (Skarin, 2010).

Kanban has been studied from different perspectives, in the context of software development. Nikitina, Kajko-Mattsson and Strale (2012) reported the transition of scrum to scrumban — the combination of scrum and Kanban approaches. Although, the transition brought positive result to the company, the authors however argue that, Kanban principles play little role in achieving the results. Contrastingly, their conclusion is not consistent with the finding of Kerzazi and Robillard (2013) which suggests that Kanban can help identify bottlenecks and improve release flow. Similarly, Ikonen, Kettunen, Oza and Abrahamsson (2010) conducted an empirical study in order to expose the sources of waste in software development using a Kanban. The sources of waste identified in software development are shown in the following figure 2, the finding strengthens the sources of waste proposed by Poppendieck and Poppendieck (2003).
To eliminate waste, involves identifying waste such as partially done work, extra processes and task switching. Kanban enhances visualization and organization of software project. However, Kanban will not prevent waste from going into the project, despite its positive impact on the project as a whole. (Ikonen et al., 2010.)

Seikola, Loisa and Jagos (2011) report the implementation of Kanban, and the key themes from the implementation are: people, backlog, Kanban-board, WIP, meeting and metrics. The benefits experienced after implementing Kanban, include; visibility, strong base for continuous improvement, improve teamwork and cooperation. However, the challenges include; unpredictable customers demand, distributed development, and hard to avoid task switching.

Ikonen (2010) investigated the relevance of leadership in a self-organizing team by exploring how types of leadership affect waste in Kanban-driven software development projects. The finding reveal that considerable amount of waste can be mitigated with a suitable style of leadership. In addition, Ikonen et al. (2011) studied the impact of Kanban on software project. The finding show that Kanban increases teams’ motivation, control over the development process and provides real-time visualization of the project. They, however, suggest more studies in industrial context. The study called for more empirical studies in “industrial setting” (p. 313).

Anderson, Concas, Lunesu, and Marchesi (2011), used a software process simulation model to investigate Lean-Kanban approach. The model handles three types of events; feature-creation— adding new features to the backlog, feature-work-ended— when a developer has completed task, feature-to-pull— involving pulling new task or previous activity. These events are sufficient to carry out the entire simulation process. Subsequently, the model was used to study both limited and an unlimited WIP processes. The finding support limited WIP approach, since in contrast to unlimited, there a constant work flow. Concas, Lunesu, Marchesi, and Zhang (2013) extended the study, using simulation model for software maintenance process. The goal was also to compare limited and an unlimited WIP. The result show that limited WIP approach increases throughput and reduces cost.

Other studies have also reported positive influence of Kanban to software development. An experience report by Neely and Stolt (2013) shows that the hybrid of scrum and
Kanban (Scrum-ban) principles can enable continuous release of software delivery. They also argue that implementing Kanban will affect all departments. Their findings show that limiting work in progress is a good control variable. Moreover, Ahmad et al. (2013) found that, companies that have implemented Kanban, experience shorter lead time. Also, Oza, Fagerholm, and Munch (2006), report that Kanban improves communication and collaboration among team members. Further, Sjøberg, Johnsen, and Solberg (2012) replaced Kanban with Scrum process, and the lead reduced, improve productivity, and improve quality. Similarly, Polk (2011) report Kanban improves process performance.

In spite of the benefits reported in applying Kanban to software development, its application still suffers several limitations. According to Sugimori et al. (1977) Kanban operates in a sequential production line, however Karvonen, Rodriguez and Kuvaja (2012) lament that having such production line in software development is difficult to achieve. Also, Wang et al (2012) also report the difficulty of adapting Kanban, thus, recommend an implementation guideline, echo by Ahmad et al. (2013). Based on these reports, more studies are required in other to solve the challenges experienced using Kanban. Thus, this study aimed to advance the knowledge of Kanban by consulting existing literature of industrial engineering.
3. Systematic literature review

The study was carried in part, using Systematic literature review research method. This chapter details the application of the SLR to the current study. First, the overview of the research method and its key stages is presented. Then, each stage is further elaborated across rest of the section. The approaches employed to reduce validity threats and limitation of the study will be discussed in chapter 7.1 validity and limitation of study.

SLR research method is a common research method within the field of medicine, but it is now being used in software engineering research as well. SLR is used to identify, evaluate and interpret all available research pertinent to a specific research question or phenomenon of interest. Further, SLR is undertaken for several reasons such as gathering available evidence about a specified treatment or technology, uncovering research gaps in existing knowledge in order to propose the areas that require further investigation. SLR can also enable the positioning of new research study. (Kitchenham, 2004.)

SLR is in contrast to the traditional literature review employed in research study. SLR is however, a method that was developed to formally and systematically gather and, appraise evidence pertaining to a phenomenon (Kitchenham 2004; Biolchini & Jorge, 2005). The central focus of SLR is to combine empirical studies on a focused idea, which will as a result, enhance generalization to a greater extent compared to using a single study. Additionally, SLR brings several advantages, it helps to summarize the existing knowledge of a particular phenomenon. It exposes any research gaps to suggest areas requiring further research and to specify a framework to aptly position novel research projects. SLR research outcome is repeatable, because all research activities should be well documented. Consequently, the rigorousness of a SLR is determined by the level of documentation details. (Kitchenham, 2004; Kitchenham & Charters , 2007.)

Software engineering field is confronted with different challenges, mainly in terms of improvement in its methodology. It is common that software developers have inadequate evidence to confirm that altering a particular software method may improve the practices. Thus, SLR research method is important in software engineering field, because it provides research syntheses that can better inform practitioners on available best practices. (Kitchenham, 2004; Kitchenham & Charters , 2007.)

The guidelines commonly use to perform SLR in software engineering field was developed by Kitchenham (2004) and Kitchenham and Charters (2007). Research studies that have used the guidelines in software engineering field include Dyba and Dingsøyr 2008, DyAfzal, Torkar and Feldt 2009, Ahmad et al 2013. The SLR process comprises of many distinct activities. (Kitchenham, 2004; 2007; Biolchini & Jorge, 2005.) The steps depicted in figure 3 which are in accordance with the guidelines of Kitchenham (2004) and Kitchenham and Charters (2007) was also adopted in the current study.
Figure 3. Stages in systematic review process, based on (Kitchenham, 2004).

3.1 Planning the review

In planning phase, the objectives of the review are stated, which will facilitate the review protocol. The review protocol defines the review research question and the strategy used in conducting the review. To efficiently define the components in the review protocol, a brief review was conducted using the industrial engineering literature. This activity enhanced the definition of search keywords and the emergence of terms like- Constant- Work-in process (CONWIP). The examples of studies reviewed at the initial stage are Kumar and Panneerselvam, 2007; Mukhopadhyay and Shanker, 2005; Anderson, 2010; Jones, 2008. In addition, this provided the researcher with the initial knowledge of Kanban in that field.

Research questions: It is obligatory to have pre-defined and focused research question(s) before conducting a SLR because it would help in developing robust review protocol. (Biolchini & Jorge, 2005.) Hence, table 3 shows the research questions for the review in this study. Details on how they were generated was explained in section 1.3.
Table 3. Research questions.

<table>
<thead>
<tr>
<th>ID</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1</td>
<td>How Kanban is described in the existing scientific literature of Industrial engineering?</td>
</tr>
<tr>
<td>RQ2</td>
<td>What types of Kanban systems exist in the scientific literature of Industrial engineering domain?</td>
</tr>
</tbody>
</table>

**Developing the review protocol:** the review protocol specifies all the activities involved in an SLR, in other words, it is a detailed plan of the review. A pre-defined review protocol is required in any SLR because it serves as a guide when conducting the review. The review protocol (see Appendix A) was reviewed and evaluated by two researchers afterwards. One from the department of Information Processing science and the other is from the industrial engineering department, University of Oulu, Finland. The following sections reflect the content of the review protocol.

### 3.2 Conducting the review

In this phase, the activities in the implementation phase shown in Figure 3 were carried out. Literally, it is the execution of the review protocol, because it is informed by the content of the review protocol.

#### 3.2.1 Data sources and search strategy

The electronic databases selected are:

- Scopus
- IEEE Xplore
- Web Of Science
- ACM

These databases were selected due to the review topic. For instance, 30% of subject covered in Scopus is physical sciences. Also, Scopus contains journals and conferences publications. ACM major in computing research, so it was included to identify studies classified in that category. IEEE Xplore was also selected to identify review topic published by IEEE or it publishing partners. Lastly, Web of science provides access to multidisciplinary research as well as provides subject specific contents.

The search terms were derived based on the initial review, since the terms used in describing or associated with Kanban in the industrial literature form the search string. Also, the addition of *operation and production* to each search string was to focus the search to articles from the industrial field. The following strings are used in searching for the relevant studies from the databases are:
1. Pull AND Kanban AND (Operation OR production)
2. Toyota Production System AND Kanban AND (Operation OR production)
3. Kanban AND (Operation OR production) AND "Inventory system"
4. Lean AND Kanban AND (Operation OR production)
5. CONWIP AND Kanban AND (Operation OR production)
6. Kanban AND (Implementation OR Benefits OR waste elimination) AND (Operation OR production)
7. just-in-time OR JIT AND Kanban

When searching Scopus database, these entire search strings were combined by using the Boolean operator “OR,” which implies that, the article needed to include at least one of the terms.

1 OR 2 OR 3 OR 4 OR 5 OR 6 OR 7.

To develop the search strategy, a Librarian in Oulu University, Finland, was contacted. The research interest was discussed, and the outcome of that meeting informed the researcher on the procedure used in searching different electronic databases, entering search terms and combining terms to form search strings. One of the researchers contacted, gave advice on the research interest and identification of the field's electronic databases. All these exercises with the preliminary study on the subject heavily influenced the search strategy and search terms used. Figure 4 shows the search strategy employed in this review.

![Flowchart showing the search strategy](image_url)

**Figure 4.** Search strategy, adapted from (Unterkalmsteiner, Gorschek, Islam, Cheng, Permadi & Feldt, 2012).
According to Kitchenham (2004), it is advisable to conduct a pilot search to validate the search strategy. If the search results are in line with the expectations, then, the actual search can be conducted, otherwise, the search strategy defined in the review protocol should be modified. At the outset, the search strategy was evaluated by piloting the inclusion and exclusion criteria. The reason for the pilot test was to validate that relevant studies relating to the research questions can be identified. At this stage, the search strategy defined in the review protocol was applied. The strategy was piloted using one of the selected electronic databases (Scopus), and its outcome was positive towards the research interest. The actual search was initiated afterwards.

### 3.2.2 Selection of primary studies

The review selected relevant studies in accordance to the inclusion and exclusion criteria. Thus, the inclusion and exclusion are listed as follows:

In the review, a study is considered relevant if it fulfilled the following criteria:

- Peer reviewed to guarantee some level of quality.
- Focused on ‘Just-in-Time’ production—according to the initial study, Kanban control system is the main tool used to achieve the objectives of JIT, so it is expected that studies addressing JIT will also discuss Kanban system.
- Focus on Kanban system—Kanban is the subject under review in this thesis.
- Published between 1977-2013. The first article published on JIT was published 1977 (Sugimori et al. 1977). This should help uncover the fundamental of a Kanban system.
- Written in English Language.

The following were used to exclude studies:

- Duplicate copy of the same paper already included.
- Papers which did not address Kanban systems.

### 3.2.3 Study selection procedure

The process employed in selecting the primary studies was divided into phases. Table 4 shows the five phases in terms of the total number of studies, after each phase.
Table 4. Steps involved in selecting primary studies.

<table>
<thead>
<tr>
<th>Phase</th>
<th>Number of studies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1: Conduct searches using the search keywords</td>
<td>1552</td>
</tr>
<tr>
<td>Phase 2: Remove duplicates studies</td>
<td>1366</td>
</tr>
<tr>
<td>Phase 3: Exclude based on title and abstract</td>
<td>1138</td>
</tr>
<tr>
<td>Phase 4: Exclude based on full text scanning</td>
<td>257</td>
</tr>
<tr>
<td>Phase 5: Exclude studies based on reading full text, assessment of quality of studies.</td>
<td>53</td>
</tr>
</tbody>
</table>

In phase 1, the selected databases were used to search for both journal articles and conference papers relating to the research questions, using specified search keywords.

In phase 2, the generated search results were exported to Refworks (a web-based reference management tool), in their individual folders. Subsequently, the folders were combined to ensure easy removal of duplicates with Refworks duplication removal tool.

In phase 3, the remaining studies after removing duplicates were exported to an Excel sheet. This action simplified some of the selection effort, for instance, selection based on title and abstract was made easy. Afterwards, studies were selected or excluded based on their title and abstract. Although, some abstracts were ambiguous, inconclusive or missing. Thus, the full texts of such studies were retrieved and skimmed. Ambiguous cases were discussed with the thesis supervisor and decisions to exclude or include were made.

In phase 4 the full texts of studies were retrieved and their respective contents were skimmed. If an article met inclusion criteria, such article was considered for the next phase, which was full text reading.

In phase 5 activities such as the selection based on reading article full text and studies’ quality assessment were performed. The aim in reading the full text was to have, in relation to the preceding phase, a significant knowledge about individual studies. This activity provided backing to the decision of whether to include or exclude studies. Also, the aim in assessing the quality of study was to include study that passed the specified minimum criteria threshold.

3.2.4 Study quality assessment

The activity here involves assessing quality of potential primary studies. It is important to assess the quality of studies, because it gives more detailed inclusion and exclusion criteria, it studies if the difference in quality explains differences in study results, it provides a means to measure relevance of each study. (Kitchenham, 2004.)

The quality of studies was assessed to ensure validity of the primary studies. Although, the selected studies were peer-reviewed ensuring some level of quality, in order to increase the reliability and validity of the current study, the quality of studies was measured. According to Kitchenham (2007), there is no consensus definition of quality.
However, in order to assess the quality and to reduce any form of bias, the author applied quality criteria checklist proposed by Kitchenham (2007) (see Appendix B) to the selected studies. The checklist assigned minimum threshold of 0.5 score to each item. The total score for all the items on the checklist is 9.0, thus the minimum threshold was 4.5. So studies below the minimum threshold were excluded at this stage, and the ones that met the threshold were included.

3.2.5 Data extraction

In this stage, data extraction form (see Appendix C) was used to extract general information about each primary study. In answering the research questions, an integrated coding was chosen for data extraction. Integrated approach involves creating a start list of codes as well as prepared to create new codes (if necessary). Apart from the flexibility offer by this approach, integrated approach is commonly used because it protects the theoretical basis of the research questions as compared to both deductive and inductive approaches. (Cruzes and Dybå, 2011.) Initial codes were created relating to the research questions. The full text read and relevant quotes were placed side by side to the initial codes. Also, during this process, initial codes were expanded according to themes produced as a result of the findings. The theme that emerged during this process is the *performance* of a Kanban system.

The data were extracted from each of the fifty-three primary studies selected for this review. The starting themes focused on the research questions and some new themes emerged during this period. The strategy employed was that, data from the primary studies relating to the themes were copied verbatim into a matrix table (see Appendix D). How the primary studies address the research questions was made visible, by mere viewing the matrix table, thus making data synthesis easy.

3.2.6 Data synthesis

The goal of the stage is to synthesize data to report the review of the primary studies. In addition, synthesis is done in order to provide answers to the research question. Data were extracted into a matrix table in accordance to the research questions, this highlighted similarities and differences between the primary studies. In short, the focus is to synthesize data collected from studies in order to report the state-of-the-art of phenomenon of interest.

With regards to qualitative synthesis, Noblit and Hare 1988 identify *reciprocal translations, refutational and lines-of-argument* as ways of synthesizing studies. A line of argument synthesis was employed because its goal of *discovering a whole among a set of parts* (Noblit & Hare, 1988, p.63) aligned with this research interest. Thus, the findings from the primary studies were documented in line with the proposed research questions. The assumption was that synthesizing the extracted data will facilitate answering the research questions. To refer individual studies, the referencing style used by Dybå and Dingsøyr (2008) was adopted.
4. Results of SLR

In total, the study identified fifty-three studies addressing different aspects of Kanban, from industrial engineering literature (See Appendix E). This section discusses the primary studies, primarily to answer the research questions.

4.1 Overview of studies

As regards the publications of the primary studies, it is shown that 47 (89%) of the studies were published in refereed journals, and 6 (11%) of the primary studies were published in conferences. In addition, the study types used in the primary studies include 20 (38%) conceptual studies, 18 (34%) simulation studies, 3 (6%) mathematical approach, 1 (2%) survey, 7 (13%) are reviews, and 4 (8%) case studies. This outcome as regard simulation studies, it indicates that simulation approach have been used extensively to investigate Kanban in industrial engineering. The complete list of all the primary studies is shown in Appendix E.

4.2 Research question 1: How is Kanban described in terms of aims, process, principles, benefits and suggested practices?

This section attempts to present outcome of the review towards answering the review research question 1. The presentation was done using the themes in the research question, in addition to the theme that emerged during the review.

4.2.1 Description and aims of Kanban

Data from the primary studies offered the rationale behind the development of Kanban and its relationship with TPS. Sugimori et al. (S1) offers a detailed explanation of Kanban in relation to TPS. S1 discloses that, TPS consists of two unique features; the first is JIT, whose aims include producing only the necessary quantity products, at the necessary time, in necessary quantity (Sugimori et al, S1, p. 553), simultaneously keeping the inventory (stock at hand) at a minimum level. The second unique feature is a system that promotes respect-for-humans with the ultimate goal of uncovering workers’ full potential, through active participation. JIT aim is to eliminate waste by effective monitoring of the inventory level, which will, as a result, uncover wastes. If the identified waste is eliminated, cost of production will be reduce and the production process improved. In addition, S5 note that JIT is concerned with either (1) ordering from outside suppliers to support production in such a way as to minimize raw materials inventory. Or (2) driving production schedules by customer demand in such a way as to minimize finished goods inventory p. 617. For JIT to achieve its objectives, it must follow certain rules, such as withdrawal by subsequent processes, one piece production and conveyance, and levelling production (Sugimori et al, S1). Golhar and
Stamm (S14), reports that, (Sugimori et al, S1) was the first article on JIT production in TPS, thus an influential study.

JIT system fulfils its objectives through Kanban (Sugimori et al, S1). The inspiration of the Kanban system was, influenced by an American supermarket replenishing model (Esparrago, S8; Huang & Kusiak, S24). The Kanban system is preferred relative to other pulling systems because it is cheaper, rapid and precise acquisition of facts, and limits surplus capacity of the preceding process. Kanban in Japanese means card. (Sugimori et al, S1.) Alternatively, Kimura and Terada (S2), and Esparrago (S8) reveal that a kanban may not necessarily be a card. It can also mean verbal instruction, a light, a flag, or even a hand signal. Further, the information on a typical Kanban card is a part name, part number, quantity designated (the size of the container) in the production process. (Kimura & Terada, S2; Bitran et al, S3) Also, the primary components of a Kanban system, according to Esparrago (S8) include: the authorization cards, standard containers, work centres, withdrawal and production Kanban posts, input areas and output areas.

According to Kimura and Terada (S2) a Kanban system is divided into; in-process and inter-process. The in-process station stores the parts that are produced or processed in a container, and that container is then kept in a designated location. When its subsequent process withdraws a part, the production Kanban is removed and then put in a Kanban box. The Kanban cards in that box are gathered at specific intervals and then put in the schedule board. These activities are repeated, and the production progresses. The inter-process activities are in a way similar to that of the in-process activities. The only difference is that, the in-process Kanban card on a particular container is replaced with an inter-process Kanban card. The card removed from the in-process will be placed in the Kanban box. Additionally, a Kanban system uses a pulling technique to keep completed task in a station until its succeeding process is fit to take in the completed part for processing or consumption (Sugimori et al, S1; Kimura & Terada, S2).

In the primary studies, numerous studies have attempted to explain the goal of a Kanban system. Huang and Kusiak (S24) notes that the key objective of a Kanban system is to deliver the material just-in-time to the manufacturing work-stations, and to pass information to preceding stage regarding what and how much to produce p.170. Im and Schonberger (S9) defines Kanban as a visual device to point out problems in production and to simulate management and line employees to improve productivity p. 56. This view is supported by S8, who informs that Kanban philosophy is to force problems to be revealed in the system, p.9. Similarly, Dyck et al. (S5) holds the view that, Kanban production control is the policeman which enforces compliance with stated objectives to identify and solve manufacturing system problems p.618. In addition, Huang and Kusiak (S24) reports that Toyota sees their Kanban control system as an ‘invisible conveyor line’ mainly, because the flow of Kanban (information flow) cut across the whole production system, including all departments.

In addition, Esparrago (S8), Farahmand and Heemsbergen (S18) reveal that a Kanban system can either be a dual-card system— both production and withdrawal Kanban cards are used, or a single-card Kanban system— only the withdrawal kanban is used.
The dual-card system is a system where the two Kanban cards are used simultaneously to implement Kanban. In the single-card Kanban system the combination of pull and push mechanism are applied. In practice, the subsequent process ordered for parts from its preceding process (pull), and the preceding process produces the parts and pushes them to a designated work centre (Esparrago, S8). Further, Huang and Kusiak (S24) identified some important conditions that would permit the functioning of both single Kanban system and dual Kanban system. For single-card Kanban system, the conditions are; little distance between two subsequent stages, fast turnover of Kanban cards, low WIP, small buffer space and rapid turnover of WIP, and synchronization between the production rate and speed of material handling. While, for the dual Kanban system, the essential conditions are: moderate distance between two phases, rapid turnover of Kanban cards, WIP buffer, and external buffer.

4.2.2 Kanban principles

The guiding principles of operating a Kanban system are reflected in the primary studies literature. Table 5 shows fundamental principles found in this review, which are used in implementing Kanban control system.

**Table 5. Principles of implementing Kanban system.**

<table>
<thead>
<tr>
<th>Principle</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level production</td>
<td>S1, S2, S24</td>
</tr>
<tr>
<td>Shun complex information on the production floor</td>
<td>S1, S2, S24</td>
</tr>
<tr>
<td>All parts must be withdrawn with a Kanban</td>
<td>S1, S2, S24</td>
</tr>
<tr>
<td>Only the parts needed in the process should be</td>
<td>S1, S2, S24</td>
</tr>
<tr>
<td>withdrawn from the preceding stage</td>
<td></td>
</tr>
<tr>
<td>Previous stage should ensure the part is of quality standard</td>
<td>S1, S2, S24</td>
</tr>
<tr>
<td>The exact quantity of parts withdrawn should be</td>
<td>S24</td>
</tr>
<tr>
<td>produced</td>
<td></td>
</tr>
</tbody>
</table>

These studies (Huang & Kusiak, S24; Sugimori et al, S1, Kimura & Terada, S2) maintain that the principles shown in Table 5 will increase the chances of operating a successful Kanban system. In addition, Savsar (S20) list conditions that must be fulfilled before a station can begin operation and they are when:

- a station is free.
- There is a production Kanban from the succeeding stage.
- There is a withdrawal Kanban attached to a full container.

The principles depicted in Table 5, when implemented ensure an improvement of production process. In general, improvements are measured by the inventory level (Im & Schonberger, S9). The inventory level can be reduced by reducing the numbers of kanbans. The assumption is that, reducing kanbans will in effect expose the problems in the production process (Im & Schonberger, S9; Im, S10; Miltenburg & Wijngaard, S13). Usually, there are many cards in the beginning, but they are reduced as different improvements are made to the production process (Dyck et al, S5).
4.2.3 The benefits of Kanban method

There are many benefits derived from applying Kanban. The commonly reported benefits include; reducing cost, improving quality and productivity (Dyck et al, S5; Gravel et al, S6; Fearson, S15; Kizilkaya & Gupta, S37; Takahashi & Nakamura, S43; Dallery & Liberopoulos, S42; Akturk & Erhun, S53). Table 6 shows the benefits of Kanban found.

**Table 6. Benefits of Kanban method.**

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduced waste</td>
<td>S14, S1, S5, S4</td>
</tr>
<tr>
<td>Reduced cost</td>
<td>S5, S6, S37, S43, S42, S53, S1, S30</td>
</tr>
<tr>
<td>Improved quality</td>
<td>S5, S6, S37, S43, S42, S53, S30</td>
</tr>
<tr>
<td>Increased productivity</td>
<td>S1</td>
</tr>
<tr>
<td>Increases employees’ participation</td>
<td>S9, S2, S6</td>
</tr>
<tr>
<td>Offered Effective use of workers and equipment</td>
<td>S53</td>
</tr>
<tr>
<td>Maximizes line efficiency</td>
<td>S52</td>
</tr>
<tr>
<td>Reduced inventory/inspection/product/administrative costs</td>
<td>S26, S30, S47</td>
</tr>
<tr>
<td>Prevents over-production</td>
<td>S13</td>
</tr>
<tr>
<td>Improves communication</td>
<td>S13, S15</td>
</tr>
</tbody>
</table>

Akturk and Erhun (S53), argues that, Kanban provides effective use of workers and equipment. Additionally, Kumar and Panneerselvam (S52) report that Kanban provides both high throughput rate and maximizes line efficiency. Kanban reduces inventory cost, inspection cost, unit product cost and administrative cost (Sriparavastu & Gupta, S26). Huang and Kusiak (S24) added that Kanban increases the speed of information exchange, it improves productivity and it provides visibility to production process. Miltenburg and Wijngaard (S13), mentioned that Kanban due to its effective control of inventory, prevents over-production. Kouri et al. (S47) reports that limiting WIP reduces the resources consumed at a given time. Similarly, Marek et al. (S30) found that controlling WIP increases quality, as a result reduces amount of reworks and financial losses significantly. Fearson (S15) argues that Kanban enhances communication and empowers employees.

Miltenburg and Wijngaard (S13), explains that the means JIT eliminates waste, is by reducing the inventory. As a result, the production process will experience problems, and after these problems are been solved, the result will be reduction in cost and quality improvement. Further, Miltenburg and Wijngaard (S13), reports that improving productivity through JIT is realized by reducing inventory. Reducing inventory is the same thing as reducing the number of Kanban cards. Reducing the number of Kanban cards further uncovers problems p.56. In addition, Kanban lowers WIP level through
decreasing the number of its cards or size of the containers (work item) Esparrago (S8). Miltenburg and Wijngaard (S13) explains that Kanban is able to improve communication because of the frequent interactions between stages. Further, Kanban increases the participation of employee (Gravel et al, S6). However, management should engage in the activities that would continually motivate the employee. Such activities include quick implementation of employee suggestions (Sriparavastu & Gupta, S26).

4.2.4 Performance of a Kanban system

Quite significant numbers of the primary studies discuss performance metrics for a Kanban system. This observation indicates that, to continually derive the benefits of implementing Kanban system, factors that may affect its performance is important as well. The primary studies present factors that affects the performance of a Kanban system. Ardalan (S23) argues that, the number of Kanban cards, priority rules, status of waiting withdrawal Kanban cards and withdrawal cycle time between two successive stages, have significant impact on a Kanban system. These claims were supported by Yang (S41), Khojasteh-Ghamari (S48), and Kumar and Panneerselvam (S52).

Table 8. Factors that affect the performance of a kanban system.

<table>
<thead>
<tr>
<th>Factors</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Kanban cards</td>
<td>S11, S12, S2, S41, S48</td>
</tr>
<tr>
<td>Priority rules</td>
<td>S23, S41</td>
</tr>
<tr>
<td>Status of waiting kanbans</td>
<td>S23</td>
</tr>
<tr>
<td>Withdrawal cycle time</td>
<td>S23</td>
</tr>
<tr>
<td>Transfer policy</td>
<td>S41</td>
</tr>
<tr>
<td>Lot sizes (work-items)</td>
<td>S53</td>
</tr>
<tr>
<td>Capacity management</td>
<td>S7</td>
</tr>
<tr>
<td>Product variation</td>
<td>S41, S44, S53</td>
</tr>
</tbody>
</table>

Similarly, Akturk and Erhun (S53), Baynat et al. (S44), and Yang (S41) inform that product variety, standardization and lot sizes (work-item) influence the performance of a Kanban system. Table 8 shows the factors found in the primary studies which can affect the performance of a Kanban system. When a stage in a production process is free, and there are many job items (cards) competing for the free stage, then, a priority rule can be used to choose the next work items (Yang, S41). Also, Yang (S41) identifies two priority rules used in selecting cards for Kanban systems. These rules are first come first sever (FCFS) and the maximum number of cards (MNC). FCFS gives priority to the Kanban card that has highest waiting time. While, MNC gives priority to the Kanban card that has the largest number of cards waiting for the same work.

The MNC priority rule slightly reduces the lead time, total WIP and the total number of interactions between stages, compared to the FCFS. Nonetheless, the FCFS rule is
chosen for its simplicity, while MNC is favored in terms of performance. Similarly, Lee (S4) compares shortest process time (SPT) and higher pull frequency (HPF) priority rules. The priority rule of SPT holds when all queuing jobs are not late. The result shows that HPF rule performed poorly as compared to SPT rule. Similar conclusion was reached by Akturk and Erhun (S53) with the comparison of SPT to FCFS.

The withdrawal cycle is the time interval used in withdrawing parts from an upstream stage. The withdrawal cycle considerably affects the customers waiting time and the total WIP in a Kanban system. The increase in withdrawal cycle increases the average customer waiting time, but, reduces the total WIP and number of trips between upstream and downstream stages. In addition, as regard product variety and product standardization, Akturk and Erhun (S53), reports that, increasing the product variety and decreasing the product standardization, reduces the performance of Kanban systems p.2879. Also, large lot sizes magnify lead time and inventory level in a process (Akturk & Erhun, S53). Lambrecht and Decaluwe (S7) posits that, the performance of a Kanban system depends on how sufficiently a company manages capacity.

The total number of lots allowed in an in-process at a given time is called the number of Kanban cards (Dyck et al, S5). The increase in number of Kanban cards reduces the lead time but increases considerably, the total WIP and number of interactions between stages (Yang, S41). As stated by Huang and Kusiak (S24) the fewer the kanbans the better the systems p.178. The sensitivity of the system is higher with few Kanban cards. Gupta and Gupta (S11), argue that calculating the number Kanban cards is extremely important p.351 due to its effect on the behavior of the system. In deciding on the number of Kanban cards in a Kanban system, Huang and Kusiak (S24) highlights the factors that can influence the decision as feed-rate, machining utilization, coefficient of variation of processing times. Also, adequate forecast of demand and estimation of lead time can inform the decision of adjusting the number of Kanban cards in a system Huang and Kusiak (S24). Further, Sugimori et al. (S1) offers the formula used in Toyota, to effectively distribute Kanbans.

$$n > DL (1 + \alpha) a$$  (1)

Where n is the total number of Kanbans, D is the rate of demand, L is the time taken for replenishment (lead-time), $\alpha$, is the safety factor (number of Kanban), and a, is the total numbers of containers (Sugimori et al, S1). Bitran et al. (S3) informs that the total number of kanban within a production process can only be altered by management. And Widyadana et al. (S49) offers a meta-heuristic approach in determining the optimal number of kanban in a station.

4.2.5 Suggested practices

The primary studies revealed suggestions on both successful operations and implementation of a Kanban system. In operating Kanban system, Miltenburg and Wijngaard (S13) outline several improvement measures that can be applied to a Kanban system. These measures are, reducing set-up times to allow the production of small lots, improving quality of products, improving process capability, and multi-skilling of staff. Sriparavastu and Gupta (S26), echoed the idea of developing multifunctional staff, but also added that quick implementation of staff improvement suggestion can contribute to
increasing their participation. Similarly, As regards implementing a Kanban system, Moeeni et al. (S25) and Miltenburg and Wijngaard (S13) report that Kanban implementation should be incremental, also, Miltenburg and Wijngaard (S13) added that the implementation should be backward from the final stage. In addition, Miltenburg and Wijngaard (S13), recommends that, workers within an organization should implement themselves p.13, because as compared to a consultant, they understand the system more. Im (S10), however, argue that JIT consultants are the key sources of information regarding the implementation of Kanban.

According to Im (S10), JIT systems are divided into three classes: A, B, and C. Class-A system applies Kanban for both inventory and production control. In class-B systems, Kanban is only used for shop-floor control. Class-C is a JIT system without a Kanban system. Further, Im (S10) outlined three prerequisites before Kanban implementation. The first is that, companies need to choose the suitable class of JIT system. This decision can be informed by the product design, demand volatility and lead time. Im (S10) affirms that, computer companies prefer class-B JIT system, because it is difficult to achieve a level production in such production environment. The second prerequisite is that companies have to determine the appropriate timing for the implementation. The third is that organizations should choose the suitable implementation method. On the other hand, Golhar and Stamm (S14) report that adopting an employee-oriented p. 659 management approach and focus on issues relating to production are critical in achieving a successful implementation.

4.3 Research question 2: What types of Kanban systems exist in industrial engineering literature?

This section answered the second research question by reporting the evolution of kanban control system. The Kanban system has indeed evolved and also it has inspired the development of similar control systems. The Kanban systems found in the existing industrial literature are: constant work in progress (CONWIP), hybrid systems, pure Kanban, generic Kanban system, extended Kanban system, and flexible Kanban. These systems are illustrated in Table 9, though the focus here is to reflect the types of Kanban systems found in the existing literature.
Table 9. Types of Kanban system.

<table>
<thead>
<tr>
<th>Kanban type</th>
<th>Characteristic</th>
<th>Paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONWIP</td>
<td>• Use backlog to dictate the part number sequence</td>
<td>S12, S21, S22, S31, S32, S50</td>
</tr>
<tr>
<td></td>
<td>• Cards are attached to all parts produced on the line against individual part number</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Work item is pushed between work stations once they have authorized card</td>
<td></td>
</tr>
<tr>
<td>The hybrid CONWIP</td>
<td>• Combination of Basestock and CONWIP</td>
<td>S27, S51, S50</td>
</tr>
<tr>
<td></td>
<td>• Release rate is equal to the demand, thus, restrict intermediate buffer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Use relatively limited space</td>
<td></td>
</tr>
<tr>
<td>The pure kanban system</td>
<td>• Orders are released when:</td>
<td>S29</td>
</tr>
<tr>
<td></td>
<td>• Kanban arrives from subsequent stage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Parts are supplied from preceding stage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Order requested have been processed</td>
<td></td>
</tr>
<tr>
<td>The generic kanban system</td>
<td>• Applicable in a dynamic environments</td>
<td>S16, S19, S17</td>
</tr>
<tr>
<td></td>
<td>• Controls WIP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Work item maybe unknown beforehand</td>
<td></td>
</tr>
<tr>
<td>The general kanban control system</td>
<td>• Hybrid of Kanban control system and basestock system</td>
<td>S33, S34, S42</td>
</tr>
<tr>
<td></td>
<td>• A free kanban must be available before work item is taken</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Applicable in dynamic environments</td>
<td></td>
</tr>
<tr>
<td>Extended kanban control system</td>
<td>• Hybrid of Kanban control system and basestock system</td>
<td>S42, S33, S39</td>
</tr>
<tr>
<td></td>
<td>• Authorization before processing work item</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Applicable in dynamic environments</td>
<td></td>
</tr>
<tr>
<td>Flexible kanban system</td>
<td>• Applicable in dynamic environments</td>
<td>S38, S35, S37</td>
</tr>
<tr>
<td></td>
<td>• Effective in minimizing daily WIP</td>
<td></td>
</tr>
<tr>
<td>Electronic kanban</td>
<td>• Applicable electronically</td>
<td>S45, S46, S47</td>
</tr>
</tbody>
</table>

It is apparent that, there are various types of Kanban systems in the existing literature of industrial engineering. These control systems are discussed subsequently.

Constant-Work-in-Progress: Spearman and Woodruff (S12) propose an alternative to the traditional Kanban system called CONWIP. Their goal of developing this system was to have a system that could retain the benefits of Kanban and be applicable in different manufacturing situations (e.g. unstable demands), as instability affects the performance of a Kanban system (Baynat et al., S44; Moeeni et al., S25). Spearman and Woodruff (S12), highlights that CONWIP is a generic Kanban control system. However, unlike Kanban system, the cards in CONWIP are attached to the container at the beginning of the production line, and they cross over the whole production. Thus, when the contents in the container are used by the end of the line, the attached card is
returned to the beginning of the line. So all job order must have a kanban before it goes into the production process, otherwise, the job will queue and wait for a card that is free.

The system is dependent on cards and these cards can also be in an electronic form. Another difference is that in a traditional Kanban system, each part produced is triggered by a Kanban card, while in CONWIP, production cards are designated to the production line, although both systems demand information directly from the final station (Gstettner & Kuhn, S21). Bonvik and Gershwin (S22) show that CONWIP system outperforms Kanban system as regard WIP level, and Huang et al. (S28) echo similar conclusion. Further, the performance of CONWIP over Kanban system has been addressed in these studies (Takahashi & Nakamura, S31; Framinan et al., S32). Also, from the standpoint of storage space Widyadana et al. (S49) argue that CONWIP control system is more preferred. Khojasteh-Ghamari (S48) compared Kanban and CONWIP, they argue that an optimized Kanban system usually outperforms CONWIP with the same level of cycle time but lower WIP. The study of Takahashi and Nakamura (S31) shows that both systems can handle a dynamic environment. However, the reactive Kanban has lower inventory level than the reactive CONWIP (Takahashi & Nakamura, S31), the study of Framinan et al. (S32) aligns with this conclusion.

The hybrid control system: Hybrid control system is a Kanban control system. The study of Bonvik et al. (S27), González-R et al. (S51), and Lavoie et al. (S50) describe it as the combination of Kanban system and the CONWIP system. The hybrid policy, behave such that the production is done by controlling the release rate to the production line equal to the demand. With this, the internal buffers are frequently empty (Bonvik et al., S27). Unlike the traditional Kanban systems which produce some parts beforehand and accumulate them in its output buffer. These parts are then available for use by the downstream station, even if there is a downtime in the upstream station. The subsequent station can continue its operations (Kimura & Terada, S2). Lavoie et al. (S50) claim that, if the cost of storage space is to be seriously considered explicitly, then hybrid provide the lowest cost of storage.

The pure Kanban system: Takahashi and Nakamura (S29) describe another type of Kanban, named pure Kanban. In the pure Kanban system, Kanbans are fixed to the buffer at the output point, and the requests are free when the material is been consumed. In addition, three types of information indicate if a part is to be consumed or not, and they include: (1) request from the succeeding stage, (2) the parts supplied from the preceding stage, availability of material to process the order.

The generic Kanban system: Chang and Yih (S16) propose a generic Kanban system, the rationale is the fact that traditional Kanban system is affected by instability in demand, indicating that Kanban performs poorly in dynamic environments. The assumptions behind the generic Kanban are that, the master schedule is unchangeable, and the production will not commence unless there is demand. In this system, order arrives at the final stage of the production line, but the final stage will not begin processing until all required parts arrive. Since the required materials are unknown at the onset, the only means to know is to request them from the preceding station. The preceding station cannot start processing unless there was a request from the succeeding station. In generic Kanban system, there are fixed Kanbans, and each is attached to a product in the system. Thus, a request can be responded to only when a Kanban is available and if not the job has to wait for a free Kanban. The study also reports that,
there are two stages in the generic Kanban and they include the Kanban acquisition stage and the real production stage. In addition, the study of Frein et al. (S19) identified some differences of the generic Kanban and the traditional Kanban. In the latter, a succeeding station collects part from the output buffer of its preceding station, reducing the waiting time, however, in the former, the waiting is required as only the Kanbans stay in the queue and not the actual job. Another difference is the information flow, the communication between two consecutive stages can be done independently in generic Kanban, but they are done simultaneously in Kanban. Hence, the number of kanbans and the size of lot affect the performance of the system (Chang & Yih, S17).

*The general Kanban control system (GKCS)*: Bollon et al. (S33) presents generalized Kanban control system. The system is the combination of Kanban system and base stock control system- a base stock is control system that is concerned with an immediate replacement of consumed material or work item to maintain stock (Geraghty & Heavey, S34). In a general kanban system, all demand received create a production order. So demands are not transferred immediately to the upstream because the demand must ensure there is free Kanban. The target in this system is to have a Kanban system and also maintain the maximum level of stock (which is achieved by base stock system). The difference is that in traditional Kanban system, all stages’ Kanbans are fixed to equivalent of finished parts (no free Kanban available). The situation is same in GKCS. However, there are free Kanbans to function as decoupling parameters between the upstream and downstream of the production line (Liberopoulos & Dallery, S39; Kotani, S45; Kumar & Panneerselvam, S52; Frein & Mascolo, S40).

*Extended Kanban Control System (EKCS):* several studies have investigated this system for example, (Liberopoulos & Dallery, S39; S42; Bollon et al., S33). Dallery and Liberopoulos (S42) proposed this type of pulling systems for multistage manufacturing environment, and reports that this system works exactly like the generalized Kanban system and the only difference is that, a work item can only be received in a stage if both production order and free Kanban are available. Liberopoulos and Dallery (S39), (S42) report that EKCS was designed to implement pull production policies, while combining Kanban control system and base stock control systems. The central idea of this system is that when the demand arrives, it instantaneously announces it to all stations, like in base stock. However, no part is made without an authorization from the downstream stages.

*Flexible Kanban System* was introduced to cope with uncertainties and planned/unplanned interruptions (Gupta et al., S38, p.1065). In this system, Kanban number judicially varies during the production process, to take care of disparities caused by changes. The system maintains a minimum level number of Kanbans attached to the process (stage). However, the system allows the addition of the Kanban to improve the system performance and can be removed when they are not needed. The only reason an extra Kanban is held in this system is to increase production flow. The benefits of this system include shorter throughput, reduced bottlenecks. The process of adding an extra Kanban to the system is called ‘release threshold’, and the process of removing them is named ‘capture threshold’. (Gupta et al., S38.) Flexible Kanban was compared to the traditional Kanban system. Flexible Kanban outperforms traditional Kanban in unexpected breakdown of material handling system (Gupta & Al-Turki, S35). Flexible Kanban system has proven to work well in the assembly lines as Kizilkaya and Gupta
(S37) implemented FKS in a disassembly line. FKS helps companies to enjoy the full benefits of JIT production (Kizilkaya & Gupta, S37; Gupta et al., S38).

Electronic Kanban: Is another type of Kanban which is recently introduced in Toyota Motor Corporations. The system uses computers and a communication device to connect Toyota to their suppliers. With the introduction of the system many advantages over the original Kanban system and they report reduced fluctuation and efficient change in number of Kanban, and faster response to demand change. (Kotani, S45.) Kanban In addition, as regard the design of an electronic kanban, Kouri et al. (S47) claims that the principles of original Kanban should be employed when designing it. Bo et al. (S46) propose a model suggesting that the model can effectively manage WIP for an electronic Kanban system. The realization of the electronic Kanban is consistent with Spencer and Larsen (S36) remark, that Kanban is able to be exploited across wide geographic distances and across organizational boundaries through the use of computer data p.216.
5. Result analysis

This section aims to examine the outcome of SLR using the existing knowledge of Kanban in software development. The analysis is done according to the theme used in presenting the outcome i.e. description of Kanban system, benefits, performance and types of Kanban systems. The analysis satisfies the research aim by highlighting both similarities and differences in knowledge as regard Kanban in software development and industrial engineering.

5.1.1 Description of Kanban system

In a Kanban system, the production process is in different stages. Each stage has an input and output area. The output of a typical stage includes the parts that have been produced in that stage for its subsequent stage. The parts consumed by the subsequent stages are replenished by their respective preceding stages (Sugimori et al, S1; Kimura & Terada, S2). Figure 4 is a representation of Kanban pulling system.

![Figure 4](image)

Figure 4. Pull system, based on (Kumar & Panneerselvam, S52).

Based on Figure 4, demand is requested by subsequent stations and the preceding stage responds by supplying the demand. This practice enabled most of the benefits perceived from using a Kanban system. This Kanban process description is somewhat consistent to the description found in software development setting. In software development,
Figure 5 shows a typical Kanban system. In this system, work items arrive to the ‘next features’ column. When there is space in the subsequent station, in this case “Making Feature Ready”, the work is moved there. At this station the work is broken into stories, which are then placed in the next column “stories ready to implement”. The activities are repeated until the work item gets to the column of “ready to release”. (Poppendieck & Poppendieck, 2010.)

Using the process description of Poppendieck and Poppendieck (2010), the work item is moved to the next stage, when there is room p. 123. This can be referred to as the subsequent stage requesting for work item from its preceding stage by creating room. Again, all demands arrive via next features column and features are developed through the systems, until it gets to the Ready to Release column. Comparing this to the Kanban process in Industrial engineering, the demand will start from the Ready to Release column and features will be pulled towards it. This process description of Kanban is consistent with the one presented by Anderson (2010). To this end, the Kanban in software development is somewhat the traditional Kanban. However, the review revealed several variants of Kanban systems, in which their development is driven by different production processes (Spearman & Woodruff, S12; Gupta et al., S38).

5.1.2 The benefits of Kanban method

Implementing Kanban usually changes the interaction between stakeholders. To reduce resistance to change, the anticipated benefits must be well articulated. (Anderson, 2010.) According to Anderson (2010), Kanban will optimize existing processes, through visualization and limiting WIP, improve software quality, improve lead time predictability, improve employee satisfaction, and providing opportunities for continuous improvements. These claims are supported by several studies— Ahmad et al. 2013, Nikitina et al. (2012). Also, the primary studies echoed the above claims. This indicates that the potentials of Kanban across production processes are consistent, even in software development process. The primary studies suggest that Kanban prevents or reduces over production (Miltenburg & Wijngaard, S13), and overproduction according to Poppendieck and Poppendieck (2003), is referred to as extra features in software development, thus it can be implied that Kanban prevents extra features which are sources of waste (Ikonen, 2010) in software development, hence a benefit.

The primary studies offer the factors contributing to some of the benefits. For instance, Miltenburg and Wijngaard (S13), states that, improving productivity through JIT is realized by reducing inventory. Reducing inventory is the same thing as reducing the number of Kanban cards. Reducing the number of Kanban cards further exposes problems and repeats the improvement procedure (Im & Schonberger, S9, p.56). Equally, Kanban lowers WIP level through decreasing the number of its cards or size of the containers. The motivation of staff through Kanban is achieved by increasing their participation. (Esparrago, S8.) Also, Huang and Kusiak (S24), reports that Kanban is responsive to the environment. And that, the benefits derived from using Kanban system comes from the environment, and not the Kanban system itself. Miltenburg and Wijngaard (S13) pointed out that JIT elimination of waste is by reducing the inventory. As a result, the production process will experience problems, and after these problems have been solved, the result will be reduction in cost and quality improvement.
5.1.3 Kanban system performance

In software development, Kanban team’s performance is measured by delivering on time and the accuracy of estimations— contrasting the estimated lead time to the actual (Anderson, 2010). Anderson (2010) states that *estimate versus actual demonstrate how predictable the team is and how well they are performing in terms of value and throughput* p.142. Throughput is similar to Agile velocity metric—it indicates how many user stories are completed within a given period. But throughput in Kanban method is used to indicate the performance of the system and illustrate continuous improvement. (Anderson, 2010.)

Specifically, the performance of software Kanban system is determined by the *quality and quantity of software being delivered, the frequency of delivery, and lead time to deliver it* (Anderson, 2010, p.178). This review revealed, the factors that can affect the throughput of a Kanban system contributing to its overall performance. Table 10 shows the factors and their corresponding translation in software development. Although, some factors could not be translated or mean the same in software development, thus retaining their name.

**Table 50.** Factors affecting the performance of a Kanban system.

<table>
<thead>
<tr>
<th>Factor in Industrial engineering</th>
<th>Factor in Software development environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Kanban cards</td>
<td>Number of Agile-Kanban cards (Larman &amp; Vodde, 2009).</td>
</tr>
<tr>
<td>Priority rules</td>
<td>Priority rules (Anderson, 2010)</td>
</tr>
<tr>
<td>Withdrawal cycle time</td>
<td>Time taken to lessen WIP limit (Poppendieck &amp; Poppendieck, 2010)</td>
</tr>
<tr>
<td>Lot sizes</td>
<td>Work item, such as user stories (Anderson, 2010)</td>
</tr>
<tr>
<td>Capacity management</td>
<td>Capacity management (Anderson, 2010)</td>
</tr>
</tbody>
</table>

Reducing the *numbers of Kanban* in a system will increase the throughput rate, since there will be fewer inventory. Also, reducing the cards optimizes a Kanban system by revealing the sources of bottlenecks and waste in a Kanban system. This could mean that, reducing the *Agile-Kanban cards* (Larman & Vodde, 2009) results in higher throughput rate because there will be fewer WIP in the Kanban system. In addition, reducing the Agile-Kanban cards could reveal sources of bottlenecks. Based on Anderson (2010) remarks on bottleneck, it is a place in the process where a *backlog of work builds up waiting to be processed* p.196. So reducing cards will quickly expose bottlenecks because there are fewer cards and the stage which is unable to cope with the development pace is taken as the bottleneck in the system.

Priority rule is an important theme in a Kanban system because it simplifies the decision to take in new work item to the system or stage (Yang, S41). Also, prioritization strategy is a key outcome of queue replenishment meetings (Anderson, 2010). This factor was identified by Anderson (2010) to have effect on the *overall lead time through the system* p.85 The primary studies disclose three of such strategies- shortest
processing time (SPT), first-come-first-sever (FCFS), and maximum number of cards MNC (Lee, S4; Yang, S41).

Further, the *withdrawal cycle time* can be translated to the *time taken to lessen WIP limit* because the former, in industrial engineering means the time interval taken to withdraw parts from an upstream stage, and every withdrawal, triggers replenishment. In software development, replenishment is triggered when a subsequent stage has at least one less than the WIP limit of that process. When this happens the completed work item in the preceding stage is transferred to the subsequent stage (Poppendieck & Poppendieck, 2010). This process is analogical to that of industrial engineering. Nevertheless, the primary studies suggested that this factor influences the throughput of a Kanban system.

The lot sizes mean work items or users stories in software development. The primary studies identified this to have effect on the throughput of the system (S5). Ambiguous users’ stories or work items may take longer time to process, if they are not appropriately sized. The effect could create bottlenecks in the Kanban system increasing the lead time for the system. The ability of an organization to manage capacity can also affect the performance of the system (Lambrecht & Decaluwe, S7). Anderson (2010) mentions the importance of capacity management in enabling an effective Kanban system. The primary studies reaffirmed this, urging organizations to develop capacity management skills to have a productive Kanban system (Lambrecht & Decaluwe, S7).

### 5.1.4 Variants of Kanban systems

The types of Kanban systems found in this review include; CONWIP (Spearman & Woodruff, S12), Hybrid (Bonvik et al., S27; González-R et al., S51), pure Kanban (Takahashi & Nakamura, S29), generic Kanban (Chang & Yih, S16), general Kanban (Bollon et al., S33; Geraghty & Heavey, S34), extended Kanban (Dallery & Liberopoulos, S42), flexible Kanban (Gupta et al., S38), and electronic Kanban (Kotani, S45). However, considering the motivation behind each system, the observation was that, production environment is ultimate driving force behind these different Kanban systems. This informs software development community on the relevance of context in operating a successful Kanban system. Software development environment is highly dynamic, to exploit the full benefits of Kanban, the need to develop context oriented Kanban system suitable in software development is eminent. In fact, Huang and Kusiak (S24) reports that, the benefits of implementing Kanban systems result from the environment, not the systems themselves p.177.

The feasibility of applying Kanban system to software distributed teams has been a reoccurring question (Anderson, 2010). According to Anderson (2010), electronic tracking should be used to support distributed teams. Electronic tracking is taken as an alternative to the card wall, which generate metrics and reports through data-gathering (Anderson, 2010). However, the electronic Kanban system revealed by this review may have more potential relatively for distributed teams. Electronic Kanban system was developed by Toyota Motor Company, specifically to synchronize their suppliers with their production process (Kotani, S45). This idea can be translated to distributed software development process, in which the *suppliers* will mean *distributed teams*.
6. Discussion

In this section, the findings are compared with the existing knowledge of software development, and research implications are highlighted. Precisely, the goal is to explicitly answer the central research question which is: *what knowledge of Kanban in industrial engineering literature can be used to improve its application in software development?*

This process description of Kanban presented by Anderson (2010) and Poppendieck and Poppendieck (2010), is consistent with the one in the primary studies. Implying that the Kanban in software development is the original Kanban (Sugimori et al, S1; Kimura & Terada, S2), as the review however, revealed several variants of Kanban systems such as CONWIP (Spearman & Woodruff, S12), Hybrid (Bonvik et al., S27; González-R et al., S51), pure Kanban (Takahashi & Nakamura, S29), generic Kanban (Chang & Yih, S16), general Kanban (Bollon et al., S33; Geraghty & Heavey, S34), extended Kanban (Dallery & Liberopoulos, S42), flexible Kanban (Gupta et al., S38), and electronic Kanban (Kotani, S45), in which their development is driven by different production processes (Spearman & Woodruff, S12).

In terms of the motivation behind each variant system the observation was that, production environment is the ultimate driving force behind these variants (Gupta et al., S38). This informs software development community on the relevance of context in developing and operating a successful Kanban system. Software development environment is dynamic (Lehman & Parr, 1976), hence to exploit the full benefits of Kanban reported in (Ahmad et al. 2013), the need to develop context oriented Kanban systems suitable in software development is eminent, as it could help understand the challenge of integrating Kanban projects with existing environments and processes (Ahmad et al. 2013), and eventually relevant in the possible development of guideline (Ahmad et al. 2013) to enhance understanding the whole Kanban process.

In addition, the idea of electronic Kanban system used in synchronizing suppliers with production process as revealed by the primary studies (Kotani, S45) may be relatively adopted for distributed software development teams, as it can be related and applied along with the electronic tracking, used to support distributed teams (Anderson, 2010).

In terms of benefits, the primary studies analysis indicates that the potentials of Kanban in software development are consistent, with those observed in the primary studies, though the primary studies reveal that Kanban prevents or reduces over production (Miltenburg & Wijngaard, S13), and over production according to Poppendieck and Poppendieck (2003), is referred to as extra features in software development, hence it can be implied that Kanban prevents extra features which are sources of waste (Ikonen, 2010) in software development, implying a perceived additional benefit. The review also reveal that, the challenges regarding motivation of employee reported by Ahmad et al. 2013 could be achieved by quick implementation of employees’ suggestions and with the use of employee oriented management approach (Sriparavastu & Gupta, S26). Frequent interaction between people involved in development stages (Miltenburg &
Wijngaard, S13) could reduce or control the issue of collaboration and communication which Ahmed et al. (2013) mentioned as one of the challenges been faced in the software development.

In terms of performance, reducing the number of cards optimizes a Kanban system by revealing the sources of bottlenecks and waste in a Kanban system (Im & Schonberger, S9). This could imply that, reducing the Agile-Kanban cards (Larman & Vodde, 2009) results in higher throughput rate because there will be fewer WIP in the Kanban system. In addition, reducing the Agile-Kanban cards could reveal sources of bottlenecks. Based on Anderson’s (2010) remarks on bottleneck, it is a place in the process where a backlog of work builds up waiting to be processed p.196. So reducing cards will quickly expose bottlenecks because there are fewer cards and the stage which is unable to cope with the development pace is taken as the bottleneck in the system, eventually resulting in reduced lead time and productivity (Ahmed et al. 2013).

Priority rule in Kanban (Lee, S4) is an important performance relate theme as it eventually has effect on the overall lead time through the system (Anderson, 2010) and primary studies reveal it can be achieved using strategies such as- SPT, FCFS and MNC (Lee, S4; Yang, S41), hence a means of achieving an even improved lead time reduction. The ability of an organization to manage capacity can also affect the performance of the system (Lambrecht & Decaluwe, S7). Anderson (2010) mentions the importance of capacity management in enabling an effective Kanban system. The primary studies reaffirmed this, urging organizations to develop capacity management skills to have a productive Kanban system.

Some of the findings mirrored existing knowledge of Kanban in software development literature. For instance, the benefits of Kanban (e.g. reduce lead time, improve quality, improve release flow, expose waste) reported in (Ahmad et al., 2013; Anderson, 2010; Ikonen et al., 2010; Kerzazi, & Robillard, 2013; Middleton, 2001; Middleton, & Joyce, 2012) were also consistent with these studies (Dyck et al, S5; Gravel et al, S6; Fearsone, S15; Kizilkaya & Gupta, S37; Takahashi & Nakamura, S43; Dallery & Liberopoulos, S42; Akturk & Erhun, S53). Effectively, articulating the benefits of Kanban can influence, positively the decision to adopt Kanban method among the stakeholders (Anderson, 2010). This study boosts these claims, by adding that it maximizes line efficiency (Kumar & Panneerselvam, S52), reduces inspection cost, reduces product cost, it self-regulation reduces administrative cost (Sriparavastu & Gupta, S26).

**Research implications:** These findings have important implications for Kanban transformation in software industry. This study provides a reaffirmation of the description of Kanban system in terms of its process, the various types and how the knowledge of Kanban from industrial engineering could be used to improve its application in software development. The following perceived implications of this study could be of importance to practitioners as well as researchers

1. The outcome of this study can inform the development of Kanban guidelines basically for software development process. This is a positive respond to the problems software development confronts regarding Kanban usage as reported in (Ahmad et al., 2013; Corona & Pani, 2013; Wang, Conboy, & Cawley, 2012).
2. The variants of the Kanban systems presented in this study can be explored further, because it may enhance in developing a detailed guideline for Kanban in software development. Since, the method used in developing them can be valuable.

3. Considering the relevance of Kanban to JIT (Lean), this study reaffirmed that Lean software development (Poppendieck & Poppendieck, 2003) is likely not to yield expected results without a Kanban system. Thus, software companies adopting Lean software development should also consider implementing Kanban system.

4. This study has offered knowledge of Kanban from the industrial engineering, using SLR (Kitchenham, 2004). Based on the relevance of the findings to software development, this study contributes to the importance SLR and interdisciplinary studies in software engineering.
7. Conclusion

In software development, Kanban system is increasingly difficult to be ignored, due to its several advantages. Its benefits include, reducing lead time, and improving software quality amongst others (Ahmad et al., 2013; Anderson, 2010; Ikonen et al., 2010; Kerzazi & Robillard, 2013; Middleton, 2001; Middleton & Joyce, 2012). Despite the growing number of studies on Kanban in software development, the suggested Kanban practices, however, have proven to be inadequate. Thus, evidence of difficulties relating to both implementation and operational issues exist (Ahmad et al., 2013; Wang, Conboy & Cawley, 2012). Part of the reason why these challenges still exist is due to the limited Kanban knowledge (Corona & Pani, 2013).

This study attempted to contribute in solving these problems, particularly, from the standpoint of limited knowledge by reviewing Kanban related literature of industrial engineering. Owing to the fact that, Kanban concept originated from industrial engineering field, thus, its maturity level in industrial engineering field is higher as compared to software engineering and in line with the suggestion of Ikonen et al. (2011). Accordingly, this study used a Systematic literature review method, following the guidelines proposed by Kitchenham (2004) and Kitchenham and Charters (2007) in exploring Industrial engineering literature.

Based on the review protocol, this study identified 53 relevant studies. The primary studies comprise of 47 (89%) refereed journals, and 6 (11%) conference proceedings. Further, with respect to the research methods used in the primary studies, 20 (38%) are conceptual, 18 (34%) are simulation models, 3 (6%) mathematical approaches, 1 (2%) survey, 7 (13%) are reviews, and 4 (8%) are case studies. These studies address various aspect of Kanban, from its description of Kanban to its types.

This study found that the process description of Kanban presented by Anderson (2010) and Poppendieck and Poppendieck (2010), is consistent with the one in the primary studies. Implying that the Kanban in software development is the original Kanban (Sugimori et al, S1; Kimura & Terada, S2), as the review reveal several variants of Kanban systems.

This study also found that various types of Kanban systems exist. The general motivation behind Kanban variants is due to the sensitivity to production environment. This is important information because it informs software development community on the relevance of context-oriented Kanban for software development. In addition, the current study found that, the first-come-first-served (FCFS), shortest process time (SPT) or maximum number of cards can enhance prioritizing decisions making in software development.

This study suggests that electronic Kanban (Kotani, S45) could be adapted to handle the issue of applying Kanban by distributed software development teams. It also reveals the added benefit of preventing extra features (Miltenburg & Wijngaard, S13). In addition, motivating staff could be achieved by quick implementation of employees’ suggestions
(Sriparavastu & Gupta, S26) and having an employee oriented management approach in place (Im & Schonberger, S9). It notes that that frequent interaction between development stages (Milstenburg & Wijngaard, S13), could reduce collaboration and communication related challenges.

Also, the evidence from this study revealed that Kanban is the strength of JIT (Lean) concept. This implies that exploring full benefits of Lean software development (Poppendieck & Poppendieck, 2003) are likely impossible without a Kanban system. Thus, practitioners adopting Lean software development method should consider implementing Kanban system.

7.1 Validity threats and limitations of study

The critical validity threat to the current study is the fact that, the author is not from the field of industrial engineering. However, this limitation was mitigated by conducting an initial study. The reason was to gain background information on the subject from industrial engineering literature. Another ultimate challenge in the study was defining the search terms, since the investigated area was unfamiliar to the author. Hence, searching within the selected discipline would have been difficult. This threat was also considerably reduced by the initial study conducted before commencing the review.

Selection bias was also a threat, but different strategies were used to reduce the impact of this threat. First, a clear and balanced search string according to the objectives of the study was used. Subsequently, a pilot search was conducted in Scopus database. The search terms were tested and thereafter validated based on the results it generated. In addition, to ensure that relevant papers were captured, the publication year range was set to 1977-2013. Interestingly, both research 1 and research question 2 were answered due to the wide search range, because both the first article on JIT and Kanban evolution were published in 1977 and 1991 respectively. This also means that, research question 2 would have been vague without the studies from 1991. Second, different electronic databases were searched including, ACM, Web of Science, IEEE Xplore and Scopus. Third, to protect against bias as results of selecting articles by screening; keywords, title, and abstract. The protocol review was evaluated by other researchers.

With respect to the research method, this study kind of deviated from the guidelines provided by Kitchenham (2004), since only one researcher selected the primary studies. However, the thesis supervisor was contacted in situation where the researcher is unsure about some studies. Also, the inclusion and exclusion criteria were evaluated by other researchers. Thus, the review included as many as possible studies based on the inclusion criteria.

Further, the review highlighted factors that could affect the performance of a Kanban system. Future studies should verify these claims within the software development process. The electronic Kanban system could be explored further, to determine its potential to support distributed software development process.
References


EUROMICRO Conference on Software Engineering and Advanced Applications, IEEE. 376-381.


Appendix A. Structure of the review protocol

1. Background

2. Research Questions

3. Search strategy
   a. search strings
   b. Electronic database to be searched

4. Study selection criteria
   a. inclusion criteria
   b. exclusion criteria

5. Study selection procedure

6. Study quality assessment checklist

7. Duplicates removal strategy

8. Data extraction strategy

9. Synthesis of extracted data
### Appendix B. Quality assessment checklist

<table>
<thead>
<tr>
<th>No.</th>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is/are the research aims clearly defined?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>2</td>
<td>Was the study designed to achieve these aims?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>3</td>
<td>Are the variables considered by study suitably measured</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>4</td>
<td>Are the data collection techniques described in detail?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>5</td>
<td>Are data collected sufficiently described?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>6</td>
<td>Is the aim of the data analysis, well defined?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>7</td>
<td>Are the study findings credible</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>8</td>
<td>Was there any negative result, presented?</td>
<td>Yes/No/Partially</td>
</tr>
<tr>
<td>9</td>
<td>Does the study present limitation, or problems that might affect the validity/reliability of the finding</td>
<td>Yes/No/Partially</td>
</tr>
</tbody>
</table>
Appendix C. Data extraction form

1. General information about the study
   a. article title
   b. article author(s)
   c. source (i.e. conference or journal)
   d. publication date

2. Specific information about the study
   a. Research method used
      i. Conceptual study
      ii. Simulation model
      iii. mathematical model
      iv. case study
      v. Literature review
   b. Research context
      i. industrial
   c. connections to research questions (Matrix)
## Appendix D. Matrix table

<table>
<thead>
<tr>
<th>Studies</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Themes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description of kanban</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aims of a kanban systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Principles of kanban system</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Types of kanban system</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Appendix E. References for the primary studies

<table>
<thead>
<tr>
<th>Study code</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study code</td>
<td>Reference</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Study code</td>
<td>Reference</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Study code</td>
<td>Reference</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Study code</td>
<td>Reference</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
</tr>
</tbody>
</table>