Systematic Literature Review of Lean Tools Applied to Software Development

Melanie Paz, Alonso Sasagawa
Universidad de Lima, Escuela de Ingeniería y Arquitectura, Carrera de Ingeniería Industrial
Lima, Perú
20171159@aloe.ulima.edu.pe, 20171456@aloe.ulima.edu.pe

José Antonio Taquía Gutiérrez
Instituto de Investigación Científica, Carrera de Ingeniería Industrial
Lima, Perú
jtaquia@ulima.edu.pe

Abstract

Currently, the predominance of Agile frameworks in software development means that Lean applications in the IT sector are incipient or very little known. However, several authors argue that the application of Lean in software has more advantages than Agile could have, due to the scalability of the procedures. The objective of this research is to know the state of the art of the application of Lean in software development through a systematic literature review. After reviewing 19 articles, it was found that the most common Lean waste with the greatest impact on software development is waiting. The main barriers that hinder the application of Lean correspond to attitudinal components of people such as low understanding of Lean concepts. The most applied Lean tool is Kanban and the main advantage of applying Lean in software development is the improvement of team performance.

Keywords
Lean, software development, lean software development, waste, systematic literature review.

1. Introduction
In the 1950s, software was developed through trial-and-error methods; some of these methods were incremental and iterative, while others were linear and sequential, such as the Waterfall model (Lei et al. 2017). This model proved to be inefficient and ineffective due to the changing requirements that could arise during software development, a characteristic given by its intangible nature. Around 2001, a group of developers proposed to change the mindset with which software was developed. This new way of thinking would be known as Agile and would emerge as a response to the Waterfall model to better deal with growing uncertainties. (Sandstø and Reme-Ness 2021). Since then, several agile frameworks, or methodologies according to some authors, have been created based on the Agile philosophy with the aim of bringing us closer to the Agile mentality. We must consider that agile frameworks such as Scrum, we manage the creation and delivery of products in short iterative cycles of time, to give in each iterative cycle with customer feedback the incrementality of value.

For the formulation of the Agile philosophy, the group of developers relied on the principles of Lean Manufacturing, applicable to the production of goods and services, adapting them to the context of software development. Lean Manufacturing is a philosophy that seeks to eliminate waste in processes. By eliminating Lean waste, productivity increases, and the organizational culture is transformed in terms of quality. (Edwin-Joseph et al. 2020).

Although the Lean philosophy is mainly applied in industrial and service sectors (Palange and Dhatrak 2021), it is recently gaining popularity in software development. Several authors, such as Kišš and Rossi (2018), Pernstål et al. (2013) or Sambinelli and Borges (2017), citing Poppendieck and Cusumano (2012), argue that the application of Lean in software development has more advantages than Agile might have, emphasizing mainly the scalability of procedures. While Agile frameworks only serve a project development level, as they were conceived for that, Lean tools are applicable to all types of processes within a company. However, at present, the predominance of agile

frameworks in software development means that the applications of the Lean philosophy in this industry are incipient or very little known.

1.1 Objectives

The objective of this research is to know the state of the art of the application of Lean in software development. For this purpose, a systematic literature review was chosen. The study will contribute to the knowledge of the IT sector by evaluating the application of Lean in software development. Research questions are:

- Q1: What are the most common Lean wastes in software development?
- Q2: What are the Lean wastes with the greatest impact on software development?
- Q3: What are the barriers to implementing Lean in software development?
- Q4: What are the most applied Lean tools in software development?
- Q5: What are the advantages of applying Lean in software development?

2. Literature Review

2.1 Lean

Lean Manufacturing is a philosophy, conceived in the Japanese automobile industry, to improve the efficiency and productivity of production processes through the elimination of waste (Shamsi and Alam 2018, Yadav et al. 2020). Waste is defined as anything that does not add value for the customer and is classified as (1) Overproduction, (2) Extra Processing, (3) Defects, (4) Transportation, (5) Waiting, (6) Motion, (7) Inventory, and (8) Non-utilized talent.
The application of Lean Manufacturing has spread to various sectors; however, there are some in which the research and application of Lean is scarce. Such is the case of the IT sector (Alahyari et al. 2019) due to the intangibility of processes and products/services offered. Given the need to eliminate waste in this sector, Lean Software Development arises, which consists of the application of the principles of Lean Manufacturing in software development. (Poppendieck and Poppendieck 2003). The principles of Lean Software Development are (1) eliminate waste, (2) amplify learning, (3) decide as late as possible, (4) deliver as fast as possible, (5) empower the team, (6) build quality in, and (7) optimize the whole (Poppendieck and Poppendieck 2003).

In view of the great differences between the industrial and IT sectors, waste cannot be classified in the same way, so it is necessary to adapt it to the nature of software (Table 1).

Table 1. Wastes of Lean Manufacturing and wastes of Lean Software Development

Wastes of Lean Manufacturing	Wastes of Lean Software Development	Definition of Wastes of Lean Software Development
Overproduction	Extra features	Features that are not requested by the customer
Extra Processing	Extra processes	Processes that don't add value (e.g. unnecessary paperwork)
	Relearning	Relearn something that we knew but have forgotten
Defects	Defects	Amount of effort, time and costs caused by fixing defects
Transportation	Task Switching	Time and effort involved in preparing for a new task
	Handoffs	Lost knowledge when work is handed off to others
Waiting	Waiting	Time lost while waiting for information, approval, materials, decisions, etc.
Motion	Motion	Unnecessary motion/effort during task execution.
Inventory	Partially done work	Work that is not completed, implemented or tested (e.g. uncoded documentation and ustested code)

Some studies have proposed ways to integrate Lean Manufacturing (or Lean Software Development) concepts into software development by co-application with other tools, practices, or ways of working. For example, Farid et al. (2017) investigated how DevOps practices improve Lean Software Development and found improvements in predictability and delivery time of 60% and 80% respectively. For their part, Birgün and Çerkezoğlu (2019) implemented a model based on Lean and Agile that resulted in time savings of 45.7% thanks to a fair distribution of the workload, efficient working conditions, and elimination of unnecessary employee movements.

2.2 Background

First, we have Kišš and Rossi (2018) who conducted a systematic literature review on the transformation from agile to Lean software development. Based on 8 articles, they determined the benefits, challenges and metrics used in the "agile to Lean" transformation process. Their results show benefits such as lead time reduction, defect correction rate improvement and process improvement. As for the challenges, these are mainly related to the internalization of Lean thinking and the most used metric is the lead time.

On the other hand, in 2013, Pernstål et al. studied Lean approaches for large-scale software development to learn about their state of the art and identify research gaps in this area of knowledge. They reviewed 38 articles in their study, all with the aim of assisting practitioners in software process improvement. They found that large-scale Lean software development research is still in its infancy. Most of the articles were not empirical and many lacked information about the context and design of the study. On the other hand, they found that the most common Lean wastes in the reviewed articles are partially done work and motion.

Likewise, we have Sambinelli and Borges (2017) who, with a broader approach, conducted a systematic review of 80 articles of Lean thinking in software engineering. With this research they sought to know the tools and practices based on Lean that have been adapted to software engineering. They also identified gaps for future research. As a result, 17 tools and 35 Lean practices were identified. The most used Lean tools in software development are Kanban, VSM and Limit Work in Progress. The most used Lean practices in the field are Continuous Integration and Test-Driven Development. The authors also point out that there are few studies related to software testing and maintenance, therefore, they represent an opportunity for future research.

Finally, Kupiainen et al. (2015) focused their study on the metrics used in Lean and agile software development. Considering 30 articles, they determined the metrics used in the field, which of these metrics have the greatest influence and the reasons and effects of metrics use in Lean and agile software development. They found 102 metrics of which the most used and most influential in software development are the metrics related to speed and effort. On the other hand, the reasons for using metrics in Lean and agile software development are the help in sprint planning, software quality measurement, project progress tracking and team motivation.

3. Methods

To answer the research questions posed in the first section, we chose to use a systematic literature review, as it will allow us to review all relevant information in an orderly fashion and with less risk of bias. According to Kitchenham and Charters (2007) this is a method for identifying and analyzing current information on a particular topic and is classified as a secondary study.

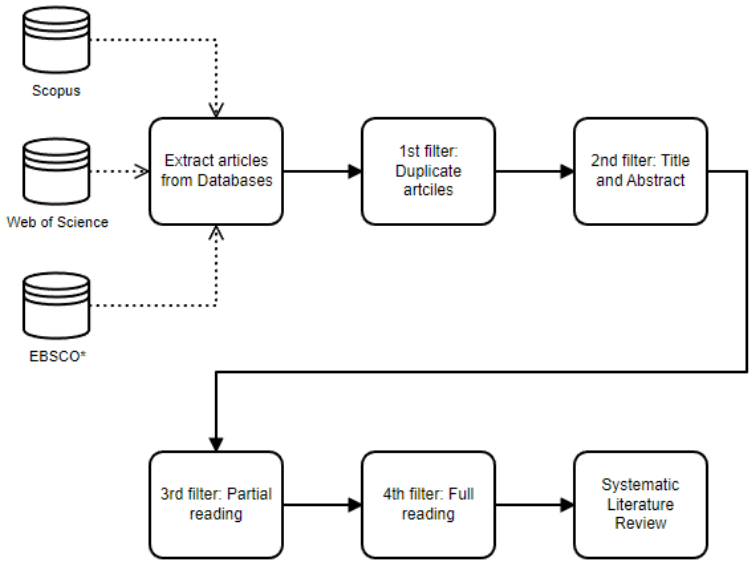

Figure 1. Research methodology

In consultation with experts, it was decided that Scopus and Web of Science would initially be used to search for articles. In the event of not reaching a quantity of at least 40 articles after the first filter, the EBSCO portal will also be used (figure 1). The search formula will be 'Lean AND ("Software development" OR "Software Engineering")', which allows us to combine the main research topics. In addition, we defined as variables to filter the articles that they must be from 2011 onwards, that only open access articles will be used, in English or Spanish languages, and that books (and their chapters) and publishers will be excluded.

For the selection of the articles to be included in the literature review, 4 filters were considered (figure 1). The first filter will be to eliminate duplicate articles. Next, they will be filtered based on their title and abstract. In the next instance, a partial reading of the article will be performed, focusing on certain aspects of the research. Finally, the fourth filter will consist of reading the research in its entirety. From this reading, a score will be given to the study based on 7 indicators about its suitability. The minimum score to be considered is 5 points.

4. Data Collection

The initial list consisted of 182 articles (considering the filters of access, language, antiquity, etc.). Of these, 34 turned out to be duplicates, so they were immediately discarded, leaving us with 148 articles.
After the second filter, we were left with 52 papers. Most of the papers discarded at this stage were because the focus of the research was either Lean or software development, but not both. For example, some articles talked about the development of an application using Lean tools but did not talk about the benefits of applying Lean in their process, because their focus was on the application itself.

After partial reading of the articles, we left 31 for total reading. Articles were discarded for various reasons. In some cases, this was because they were work in progress or proposed working models that had not yet been applied. As in the previous filter, we also found articles that used Lean principles for application development, but the focus of the research was on the application and not on the use of Lean. Finally, we found an article that had been published twice, but under different names and in different years, so it was also discarded.

Finally, after the total reading of the articles, it was decided that the literature review would be carried out with 19 of them. The selection criterion in this case was the score obtained by each article, discarding those that did not reach a minimum of 5 (table 2 and table 3). In addition, the total reading of the research allowed us to observe that we had 2 articles (F8 and F20) that, although they were not duplicates, were research that had had an extension. In other words, they were the same research, but rewritten a few months or years later. Following this finding, it was decided to consider only the most current versions of these articles (F21 and F27, respectively). The following tables show the criteria used and the scores assigned to each article.

Table 2. Qualification criteria

Code	Qualification criteria
C1	The research summary shows the problem, objectives, methodology, and the most relevant results and conclusions.
C2	Describe the observed problem from which the research starts and explain why it should be solved.
C3	Clearly define the objective of the research.
C4	The theoretical framework is presented, which contains important bibliographical references. It determines the variables and scope of the research.
C5	The methodology explains in detail the methods and techniques used.
C6	Expose the results completely.
C7	The discussion compares the results in relation to similar past research.

Table 3. Resulting scores of the fourth filter

Article code	Authors	Excellent = 1, Fair = 0.5, Deficient = 0							Total
		C1	C2	C3	C4	C5	C6	C7	
F1	Wang et al. (2012)	1	1	1	1	0.5	1	0.5	**6**
F2	Birgün and Çerkezoğlu (2019)	1	1	0.5	1	1	1	0.5	**6**

Article code	Authors	Excellent = 1, Fair = 0.5, Deficient = 0							Total
		C1	C2	C3	C4	C5	C6	C7	
F3	Karvonen et al. (2012)	1	1	1	1	0.5	1	1	**6.5**
F4	Gaete et al. (2021)	1	0.5	1	1	1	1	0.5	**6**
F5	Puspitasari et al. (2020)	1	0.5	0	0.5	0.5	0.5	0	**3**
F6	Pantiuchina et al. (2017)	0.5	0.5	1	0.5	1	0.5	0.5	**4.5**
F7	Rodríguez et al. (2013)	1	1	1	1	1	1	1	**7**
F8	Signoretti et al. (2020)								**0**
F9	Salleh and Nohuddin (2019)	0.5	0.5	1	1	0.5	0.5	0.5	**4.5**
F10	ArunKumar and Dillibabu (2016)	1	1	1	1	1	1	1	**7**
F11	Alazzam et al. (2021)	0	0.5	0.5	0.5	1	1	1	**4.5**
F12	Vallon et al. (2019)	0.5	1	1	1	1	1	0.5	**6**
F13	Cvetkovic et al. (2017)	1	1	1	1	0.5	1	0.5	**6**
F14	Neelu and Kavitha (2021)	1	0.5	1	1	1	1	1	**6.5**
F15	Fitzgerald et al. (2014)	1	1	1	1	1	1	1	**7**
F16	Fagerholm and Pagels (2014)	1	1	0.5	1	1	1	1	**6.5**
F17	Khurum et al. (2014)	0.5	0.5	1	1	1	1	1	**6**
F18	Ali et al. (2016)	1	1	0.5	1	1	1	1	**6.5**
F19	Turner and Lane (2013)	0.5	0.5	1	0.5	1	0.5	0.5	**4.5**
F20	Fagerholm et al. (2014)								**0**
F21	Zorzetti et al. (2022)	1	1	1	1	1	1	1	**7**
F22	Ahmad et al. (2016)	1	1	0	0	0.5	1	1	**4.5**
F23	Ahmad et al. (2018)	1	1	0.5	1	1	1	1	**6.5**
F24	Middleton and Joyce (2012)	0.5	0.5	1	1	1	1	0.5	**5.5**
F25	Petersen and Wohlin (2011)	0.5	0.5	0.5	1	0.5	0.5	1	**4.5**
F26	Cawley et al. (2011)	0	0.5	1	0.5	0.5	1	1	**4.5**
F27	Fagerholm et al. (2015)	1	1	1	1	1	1	1	**7**
F28	Edison et al. (2016)	0.5	0.5	0.5	1	0.5	0.5	1	**4.5**
F29	Kaushik et al. (2016)	0.5	1	1	1	1	1	0.5	**6**
F30	Nidagundi and Novickis (2017)	0	0.5	0.5	0.5	0.5	0.5	0.5	**3**
F31	Rose et al. (2021)	1	1	1	0.5	1	0.5	1	**6**

5. Results and Discussion

In this section, we will present the results of the systematic literature review, starting with the results obtained in the VOSviewer software from the keywords and based on the research questions posed in the first part of the paper.

The bibliometric information of the selected articles was entered into this software. From the map generated (figure 2), we can observe that none of the central themes of this research (Lean and software development) turns out to be the most recurrent. Instead, we find software design, this could be indicative that Lean is more applied to this branch of the computer industry. Another word that stands out in the graph is Agile, doing it in different ways: agile software development, agile methodologies, agile project management, agility, etc. A possible explanation for this is that the articles reviewed mention the application of this mindset before implementing Lean in their software development processes. Finally, we note that the only Lean tool that stands out is Kanban, a result that is shared with one of the research questions.

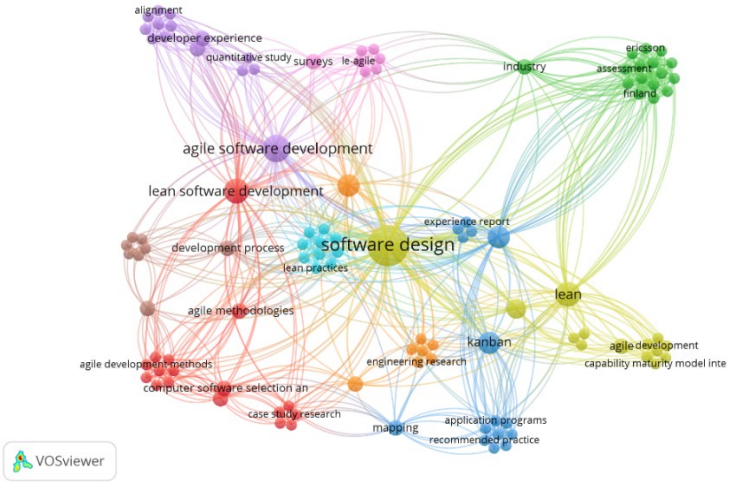

Figure 2. Keyword map generated by VOSviewer

Regarding the visualization by years (figure 3), we notice that the trend of articles is changing from research on software development in specific industries since 2012, then by software design and Lean application around 2016, to agile software development in recent years.

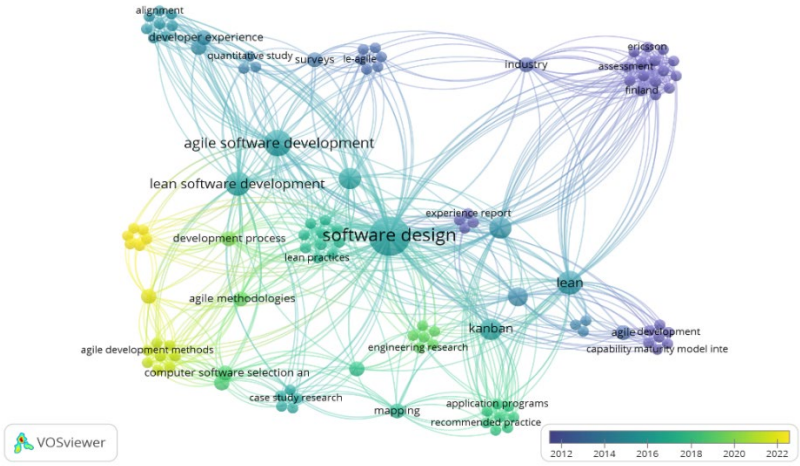

Figure 3. Map of years generated by VOSviewer

Q1: What are the most common Lean wastes in software development?

From the systematic literature review, we obtained that the most frequent Lean waste in software development is waiting with 18 occurrences, followed by defects with 12 occurrences (figure 4). It is worth mentioning that the waste of task switching/handoffs was not identified in any of the reviewed articles. To a large extent, the articles mention that waiting is due to poor time estimation and work overload; furthermore, in the case of defects, these are related to organizational pressure also due to poor time estimation (as delivery dates are usually tighter).

An example of this can be seen in the article by Cvetkovic et al. (2017) where principles of the Lean philosophy are applied to improve the company's development cycles. After the improvements are applied, the authors emphasize how limiting work in progress (WIP) and constant prioritization of the backlog (both are Lean principles) manage to increase delivery speed and decrease errors. These improvements are due to the avoidance of bottlenecks and the pressure relieved by having more realistic delivery dates.

The result obtained in this research question differs from the study by Pernstål et al. (2013) who found that the most common Lean wastes in software development were partially done work and motion. However, analyzing the nature of the industry, these wastes contribute to the generation of waiting.

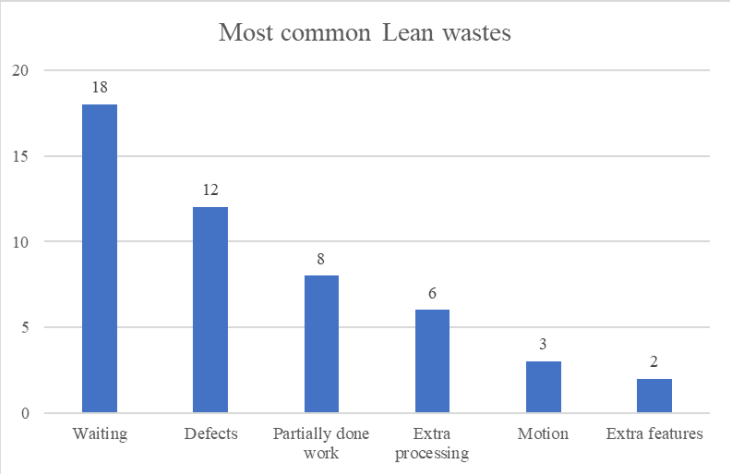

Figure 4. Most common Lean wastes in software development

Q2: What are the Lean wastes with the greatest impact on software development?
On the other hand, it was found that the Lean waste with the greatest impact on software development is delays, and by a wide margin (figure 5). Neither the waste of task switching/handoffs was identified, as in the previous research question, nor the waste of extra features.

In the study conducted by Fitzgerald et al. (2014) the authors used the Erlang-C model to estimate the number of developer teams the company should have. They found that they did not have the required capacity to meet the incoming projects, so their teams were always saturated. This situation causes delays, generating a snowball effect, because when delivery dates are not met, resources are reallocated (Lean waste) in an attempt to meet them, neglecting other projects with upcoming deadlines.

It is understandable that the waste of waiting has a big impact on software development since these projects prioritize "fast and continuous delivery to the customer" (which is a principle of both Agile and Lean Software Development). Also, for this reason, it makes sense that waiting turns out to be the most common and impactful waste in software development. It should be noted that this point is new with respect to past research since none of it considers the impact of Lean waste in the IT sector.

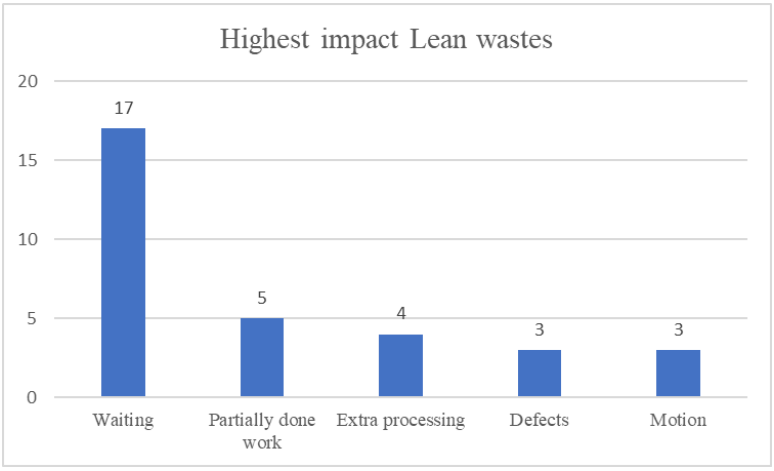

Figure 5. Lean wastes with the greatest impact on software development

Q3: What are the barriers to implementing Lean in software development?

Several barriers that hinder the implementation of the Lean philosophy in software development were found in the research reviewed. The most significant were low understanding of Lean concepts, difficulty in maintaining a disruption-free flow, and resistance to change; these were closely followed by lack of leadership (table 4 and figure 6). It is interesting to note that three of these four factors mentioned correspond to attitudinal components of people, while only one of them corresponds to process management.

Precisely, Fagerholm and Pagels (2014), conduct a study to know the Lean and Agile value structure of developers. In this study, they found that, for both philosophies, programmers and related professionals do not always fully understand the purpose of the principles (Lean and Agile) or avoid modifying their ways of working. This situation is reflected by contradictions in the responses of the teams evaluated. To mention one example, respondents claim to value the flexibility of the process, but also prefer to adhere to a carefully planned work plan.

Kišš and Rossi (2018), investigated this same point and obtained that the main barrier when implementing Lean in software development is the difficulty of workers to adapt to Lean thinking. Therefore, our results are aligned to their research. Additionally, Kišš and Rossi identified another barrier which is the difficulty to identify the "true waste".

Table 4. Barriers to implementing Lean in software development

Code	Description	Quantity
B1	Low comprehension of Lean concepts	5
B2	Difficulty to maintain process flow	5
B3	Change resistance	5
B4	Lack of leadership	4
B5	High dependency on the team	3
B6	Not having expert users in the team	3
B7	Local optimization of processes	3
B8	Lack of commitment from the team and top management	2
B9	Ambiguous customer requirements	2
B10	Other	6

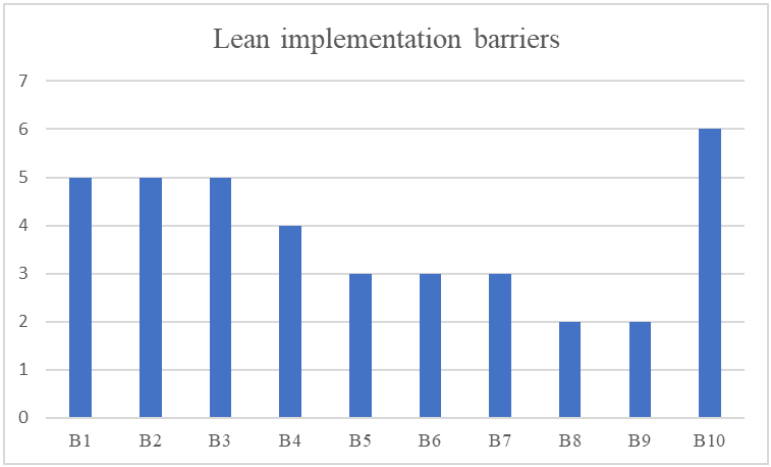

Figure 6. Barriers to implementing Lean in software development

Q4: What are the most commonly applied Lean tools in software development?

As for the most applied Lean tools, the one that was mentioned most often was Kanban. Next, we have Value Stream Mapping (VSM) and then Kaizen, also called Continuous Improvement (figure 7). This could be due to the easy

adaptation of these tools to any type of project, especially Kanban. Several authors, such as Gaete et al. (2021), Cvetkovic et al. (2017) or Ahmad et al. (2018b) mention the benefits of applying this tool in software development projects. These include improved visibility of tasks, limitation of work in progress (WIP), improved communication, etc. In the column called "others" there are tools such as 5S, Andon, Jidoka, etc. They were classified in this group since they were only mentioned once in the reviewed articles.

In the results of the study by Sambinelli and Borges (2017) it is observed that the most used Lean tools in software development are Kanban, VSM and Limit Work In Progress. This coincides with our results, with the exception of Limit Work In Progress, however, this tool is listed as the fourth most mentioned tool as well as Root cause analysis.

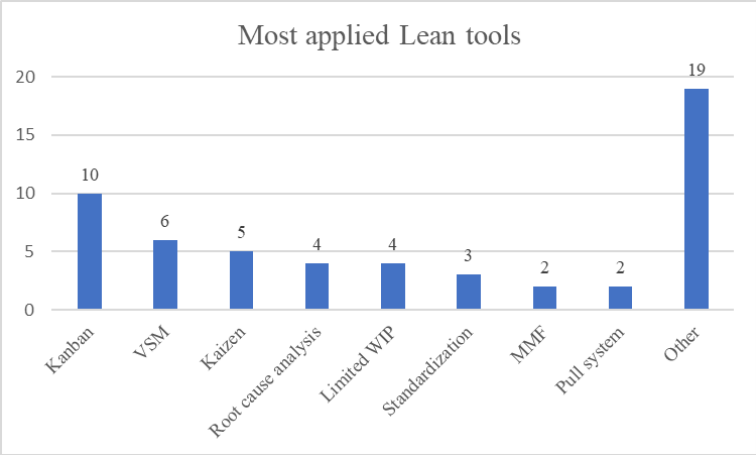

Figure 7. Most applied Lean tools in software development

Q5: What are the advantages of applying Lean in software development?

Regarding the advantages of implementing Lean in software development, it was found that the most important is the improvement in team performance. It is followed by the improvement in communication, distribution and prioritization of tasks and improvement in the organizational climate (table 5 and figure 8). Similarly, in the case of implementation barriers, it is interesting to note that three of the four major advantages are related to attitudinal components of the workers.

In a study by Neelu and Kavitha (2021) the advantages of applying a model based on Lean are evidenced. In this model the Kaizen tool is applied, and the results show improvements in several parameters after its implementation. A shorter duration per sprint, less working hours, higher productivity, fewer defects, more lines of code in less time, etc. were obtained. These results reflect the improvement in team performance.

Kišš and Rossi. (2018) also identified the advantages of applying Lean in software development. In their results they mention lead time reduction, defect correction rate improvement and process improvement. Unlike our results, these advantages are related to processes. However, we consider that the advantages identified by these authors are the result of the advantages that Lean generates in people, which were identified in this study.

Table 5. Advantages of Lean implementation in software development

Code	Description	Quantity
V1	Improved team performance	19
V2	Improved communication and team cooperation	13
V3	Efficient distribution and prioritization of tasks	7
V4	Improvement in working conditions and organizational culture	7
V5	Defect reduction	6
V6	Waiting time reduction	5
V7	Improved customer satisfaction	5

V8	Improvement in value chain management	3
V9	Better workflow visibility	3
V10	Other	2

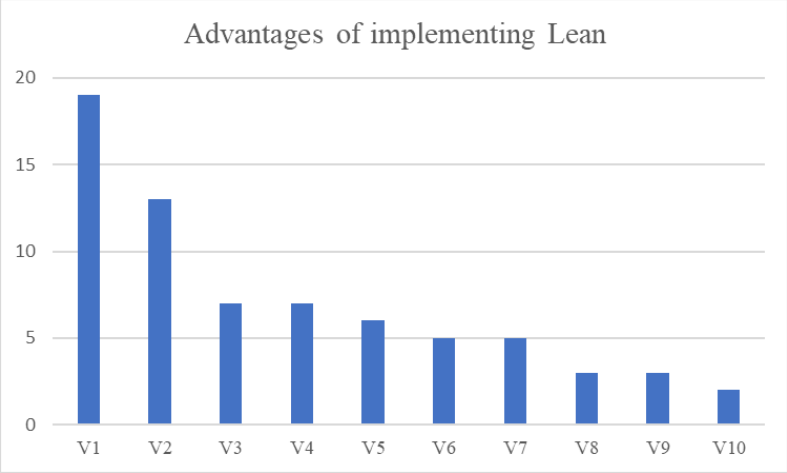

Figure 8. Advantages of implementing Lean in software development

6. Conclusion

The objective of this research was to know the state of the art of Lean application in software development. For this purpose, we focused on finding out the most common and most impacting Lean wastes, the barriers and advantages of Lean implementation and the most used Lean tools in this industry. In order to answer the questions posed, a systematic literature review was carried out, using the Scopus and Web of Science databases.

Our research found that waiting is the most common and most impactful Lean waste, presumably because of current paradigms in software development. Also, the main barriers and advantages were related to people. In terms of barriers, resistance to change and poor understanding of Lean concepts were found. On the other hand, as for the advantages, the improvement in performance and the improvement in communication and cooperation of the team were found. Finally, the most used Lean tool in software development is Kanban, a result that we attribute to the simplicity of its application and high performance for project management.

In relation to past research, we found similarities in the implementation barriers and the most used Lean tools. We also found a possible cause-effect relationship with respect to the advantages that Lean provides to software development. Finally, we obtained differences with the most common Lean wastes that our predecessors found. The question regarding the Lean waste with the greatest impact is a novelty of our study.

Limitations and future work

The main limitation of this study is subjectivity. Despite the authors' best efforts to reduce bias, it is necessary to recognize that the protocol for the systematic review of the literature admits its own criteria for the segregation of articles. For this reason, results may vary in replications of this research.
In terms of future research, one of the findings, which we highlight the most, is that the main barriers and advantages are strongly related to attitudinal or personal components. A hasty hypothesis is that since Lean is a way of thinking or philosophy, it needs to be internalized in the people who will apply it to be successful. Future research will be needed to answer this.

References

Ahmad, M., Markkula, J. and Oivo ,M., Insights into the perceived benefits of Kanban in software companies: Practitioners' views, 1*7th International Conference on Agile Processes in Software Engineering and Extreme Programming*, pp. 156-168, Edinburgh, Scotland, May 24-27, 2016.

Ahmad, M., Dennehy, D., Conboy, K. and Oivo, M., Kanban in software engineering: A systematic mapping study, *Journal of Systems and Software*, vol. 137, pp. 96–113, 2018.

Alahyari, H., Gorschek, T. and Svensson, R., An exploratory study of waste in software development organizations using agile or lean approaches: A multiple case study at 14 organizations, *Information and Software Technology*, vol. 105, pp. 78–94, 2019.

Alazzam, M., Alassery, F. and Almulihi, A., Development of a Mobile Application for Interaction between Patients and Doctors in Rural Populations, *Mobile Information Systems*, vol. 2021, 2021.

Ali, N., FLOW-assisted value stream mapping in the early phases of large-scale software development, *Journal of Systems and Software*, vol. 111, pp. 213–227, 2016.

ArunKumar, G. and Dillibabu, R., Design and Application of New Quality Improvement Model: Kano Lean Six Sigma for Software Maintenance Project, *Arabian Journal for Science and Engineering*, vol. 41, no. 3, pp. 997–1014, 2016.

Birgün, S. and Çerkezoğlu, B., A Systematic Approach for Improving the Software Management Process, *International Journal of Innovation and Technology Management*, vol. 16, no. 4, pp. 1–28, 2019.

Cawley, O., Richardson, I. and Wang, X., Medical device software development - A perspective from a lean manufacturing plant, *Communications in Computer and Information Science*, vol. 155, pp. 84 - 96, 2011.

Cvetkovic, N., Morača, S., Jovanović, M., Medojević, M. and Lalić, B., Enhancing the agility and performances of a project with lean manufacturing practices, *Proceedings of the 28th DAAAM International Symposium*, pp. 661–670, Zadar, Croatia, November 8-11, 2017.

Edison, H., Wang, X. and Abrahamsson, P., Product Innovation through Internal Startup in Large Software Companies: A Case Study, *42nd Euromicro Conference on Software Engineering and Advanced Applications*, pp. 128 - 135, Limassol, Cyprus, August 31 - September 2, 2016.

Edwin Joseph, R., Kanya, N., Bhaskar, K., Francis Xavier, J., Sendilvelan, S., Prabhahar, M., Kanimozhi, N. and Geetha, S., Analysis on productivity improvement, using lean manufacturing concept, *Materials Today: Proceedings*, vol. 45, pp. 7176–7182, 2020.

Fagerholm, F., Ikonen, M., Kettunen, P., Münch, J., Roto, V. and Abrahamsson, P., How do software developers experience team performance in Lean and Agile environments?, *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, pp. 1-10, Londom, UK, May 13-14 ,2014.

Fagerholm, F. and Pagels, M., Examining the structure of lean and agile values among software developers, *Agile Processes in Software Engineering and Extreme Programming: 15th International Conference*, pp. 218–233, Rome, Italy, May 26 – 30, 2014.

Fagerholm, F., Ikonen, M., Kettunen, P., Münch, J., Roto, V. and Abrahamsson, P., Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments, *Information and Software Technology*, vol. 64, pp. 132–147, 2015.

Farid, A., Helmy, Y. and Bahloul, M., Enhancing Lean Software Development by using Devops Practices, *International Journal of Advanced Computer Science and Applications*, vol 8, no. 7, pp. 267–277, 2017.

Fitzgerald, B., Musiał, M. and Stol, K., Evidence-based decision making in lean software project management, *36th International Conference on Software Engineering Companion 2014 - Proceedings*, pp. 93–102, Hyderabad, India, May 31 – June 7, 2014.

Gaete, J., Villarroel, R., Figueroa, I., Cornide-Reyes, H. and Muñoz, R., Enfoque de aplicación ágil con Scrum, Lean y Kanban [Agile application approach with Scrum, Lean and Kanban], *Revista chilena de ingeniería*, vol. 29, no. 1, pp. 141–157, 2021.

Karvonen, T., Rodriguez, P., Kuvaja, P., Mikkonen, K. and Oivo, M., Adapting the lean enterprise self-assessment tool for the software development domain, *Proceedings - 38th EUROMICRO Conference on Software Engineering and Advanced Applications*, pp. 266–273, Izmir, Turkey, September 5 – 8, 2012.

Kaushik, S., Avasthi, V. and Bharadwaj, A., Scrutinizing Lean Thinking and Agile Methodologies from practitioner's point of view, *Indian Journal of Science and Technology*, vol. 9, no. 20, 2016.

Khurum, M., Petersen, K. and Gorschek, T., Extending value stream mapping through waste definition beyond customer perspective, *Journal of Software: Evolution and Process*, vol. 26, no. 12, pp. 1074–1105, 2014.

Kišš, F. and Rossi, B., Agile to lean software development transformation: A systematic literature review, *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems*, pp. 969–973, Poznan, Poland, September 9-12, 2018.

Kitchenham, B., Guidelines for performing Systematic Literature Reviews in Software Engineering, Available: https://www.researchgate.net/publication/302924724, 2007.

Kupiainen, E., Mäntylä, M. and Itkonen, J., Using metrics in Agile and Lean software development - A systematic literature review of industrial studies, *Information and Software Technology*, vol. 62, no. 1, pp. 143–163, 2015.

Lei, H., Ganjeizadeh, F., Jayachandran, P. and Ozcan, P., A statistical analysis of the effects of Scrum and Kanban on software development projects, *Robotics and Computer-Integrated Manufacturing*, vol. 43, pp. 59–67, 2017.

Middleton, P. and Joyce, D., Lean software management: BBC worldwide case study, *IEEE Transactions on Engineering Management*, vol. 59, no. 1, pp. 20–32, 2012.

Neelu, L. and Kavitha, D., Estimation of software quality parameters for hybrid agile process model, *SN Applied Sciences*, vol. 3, no. 3, 2021.

Nidagundi, P. and Novickis, L., Towards utilization of lean canvas in the DevOps software, *11th International Scientific and Practical Conference on Environment. Technology. Resources*, pp. 107-111, Rezekne, Latvia, June 15-17, 2017.

Palange, A. and Dhatrak, P., Lean manufacturing a vital tool to enhance productivity in manufacturing, *Materials Today: Proceedings*, vol. 46, pp. 729–736, 2021.

Pantiuchina, J., Mondini, M., Khanna, D., Wang, X. and Abrahamsson, P., Are Software Startups Applying Agile Practices? The State of the Practice from a Large Survey, *18th International Conference on Agile Processes in Software Engineering and Extreme Programming*, pp 167–183, Cologne, Germany, May 22-26, 2017.

Pernstål, J., Gorschek, T. and Feldt, R., The Lean Gap: A Review of Lean Approaches to Large-Scale Software Systems Development, *Journal of Systems and Software*, vol. 86, no. 11, pp. 2797–2821, 2013.

Petersen, K. and Wohlin, C., Measuring the flow in lean software development, *Software - Practice and Experience*, vol. 41, no. 9, pp. 975-996, 2011.

Poppendieck, M. and Poppendieck, T., *Lean software development: an agile toolkit*, Adison-Wesley Professional, Boston, 2003.

Puspitasari. D., Pramudhita. A., Rosiani. U. and Rahmawati T., Application development of website content filling in the Information Technology Department of State Polytechnic of Malang, *1st Annual Technology, Applied Science, and Engineering Conference*, East Java, Indonesia, August 29–30, 2019.

Rodríguez. P., Mikkonen. K., Kuvaja. P., Oivo. M. and Garbajosa J., Building lean thinking in a telecom software development organization: Strengths and challenges*, International Conference on Software and Systems Process*, pp. 98 - 107, California, USA, May 18-19, 2013.

Rose, C., Nichols, T., Hackner, D., Chang, J., Straube, S., Jooste, W., Sawe, H. and Tenner, A., Utilizing lean software methods to improve acceptance of global ehealth initiatives: Results from the implementation of the basic emergency care app, *JMIR Formative Research*, vol. 5, no. 5, 2021.

Salleh, N. and Nohuddin, P., Comparative study between lean six sigma and lean-agile for quality software requirement, *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 12, pp. 212-218, 2019.

Sambinelli, F. and Francisco Borges, M., Lean Thinking in Software Engineering: A Systematic Review, *International Journal of Software Engineering and Applications*, vol 8, no. 3, pp. 15–32, 2017.

Sandstø, R. and Reme-Ness, C., Agile Practices and Impacts on Project Success, *Journal of Engineering, Project, and Production Management*, vol. 11, no. 3, pp. 255–262, 2021.

Shamsi, M. and Alam, A., Exploring Lean Six Sigma implementation barriers in Information Technology industry, *International Journal of Lean Six Sigma*, vol. 9, no. 4, pp. 523–542, 2018.

Signoretti, I., Salerno, L., Marczak, S. and Bastos, R., Combining User-Centered Design and Lean Startup with Agile Software Development: A Case Study of Two Agile Teams, *21st International Conference on Agile Software Development*, pp. 39 - 55, Copenhagen, Denmark, June 8-12,2020.

Turner, R. and Lane, J., Goal-Question-Kanban: Applying lean concepts to coordinate multi-level systems engineering in large enterprises, *11th Annual Conference on Systems Engineering Research*, pp. 512 - 521, Georgia, USA, March 19-22, 2013.

Vallon, R., Strobl, S., Ras, M., Bernhart, M. and Grechenig, T., Distributed kanban with limited geographical distance: Analyzing lean principles pull, work in progress and kaizen, *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering*, pp. 210–217, Crete, Greece, May 4-5, 2019.

Wang, X., Conboy, K. and Cawley, O., "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development, *Journal of Systems and Software*, vol. 85, no. 6, pp. 1287–1299, 2012.

Yadav, R., Mittal, M. and Jain, R., Adoption of lean principles in software development projects, *International Journal of Lean Six Sigma*, vol. 11, no. 2, pp. 285–308, 2020.

Zorzetti, M., Signoretti, I., Salerno, L., Marczak, S. and Bastos, R., Improving Agile Software Development using User-Centered Design and Lean Startup, *Information and Software Technology*, vol. 141, 2022.

Biographies

Melanie Paz is a final year undergraduate student in Industrial Engineering at Universidad de Lima, Peru. Her specialization area is modeling and optimization of processes and quality management. She did an internship at Universidad de Lima for a year. Her research interests include optimization or processes, lean thinking with focus on continuous improvement, quality management and project management.

Alonso Sasagawa is a final year undergraduate student in Industrial Engineering at Universidad de Lima, Peru, specialized in technology innovation and process modeling and optimization. He is currently doing an internship in the Computer Integrated Manufacturing Laboratory within the University, working with automation and digital fabrication technologies. His research interests include automation, process optimization and engineering design.

José Antonio Taquía is a Doctoral Researcher from Universidad Nacional Mayor de San Marcos and holds a Master of Science degree in Industrial Engineering from Universidad de Lima. He is a member of the School of Engineering and Architecture teaching courses on quantitative methods, predictive analytics, and research methodology. In the private sector he was part of several implementations of technical projects including roles as an expert user and in the leading deployment side. He worked as a senior corporate demand planner with emphasis on the statistical field for a multinational Peruvian company in the beauty and personal care industry with operations in Europe and Latin America. He is also a member of the Scientific Research Institute at Universidad de Lima. His main research interests are on statistical learning, predictive analytics, and industry 4.0.