LEAN THINKING IN SOFTWARE ENGINEERING: A SYSTEMATIC REVIEW

Fernando Sambinelli[1] and Marcos Augusto Francisco Borges[2]

[1]Federal Institute of Education, Science and Technology of São Paulo, Hortolândia, Brazil
[2]School of Technology, University of Campinas, Limeira, Brazil

ABSTRACT

The field of Software Engineering has suffered considerable transformation in the last decades due to the influence of the philosophy of Lean Thinking. The purpose of this systematic review is to identify practices and approaches proposed by researchers in this area in the last 5 years, who have worked under the influence of this thinking. The search strategy brought together 549 studies, 80 of which were classified as relevant for synthesis in this review. Seventeen tools of Lean Thinking adapted to Software Engineering were catalogued, as well as 35 practices created for the development of software that has been influenced by this philosophy. The study provides a roadmap of results with the current state of the art and the identification of gaps pointing to opportunities for further research.

KEYWORDS

Lean Thinking, Lean IT, Agile, Software Engineering, Software Development, Systematic Review

1. INTRODUCTION

Lean has been studied by researchers for almost half a century [1]. Publications in this field have increased considerably with an increasing number of industries trying to adopt this philosophy of work in its productive processes[2].

The term Lean emerged in the early 1980s when a quality gap was observed between Western and Japanese products, being most clearly noticed in the car manufacturing industry. The most comprehensive research on the differential features of Japanese productive methods was conducted by the International Motor Vehicle Program (IMVP) at the Massachusetts Institute of Technology (MIT) [3]. IMVP studies in the car manufacturing industry highlighted that Japanese companies were offering higher quality products at lower costs, compared to Western companies. This was laid down to a fundamentally different operating paradigm used at Toyota. This approach carried out by the Japanese automobile manufacturing was classified by the authors as Lean Production.

Later, Womak and Jones [4] generalized the Lean beyond production, grouping their concepts into five principles: determining what value is for the customer, mapping the value stream, establishing a continuous flow, deploying the pulled system, and aiming for perfection. This generalization was called Lean Thinking (LT) and is related to a set of applications success cases in sectors such as: hospital management, production, food distribution, building sector and services in general [1].

The process of expanding the LT to several segments also branched out into the area of information technology (IT), being called Lean IT. Within the LT application in the IT field, the

DOI: 10.5121/ijsea.2017.8302

field of Software Engineering (SE) has been considerably transformed in the last two decades as a result of this influence of thought [5]. Poppendick and Cusumano[6]describe that similarities between Japanese management and software development began to become apparent around the 1990s, following the example of Microsoft's philosophy of making daily corrections. This procedure can be seen as something similar to just-in-time, one of Lean's bases. Bell and Orzen[7] present LT applied to SE through agile methodologies (AM). They claim that AMs are a set of tools and methods for managing the production life cycles of software focusing on just-in-time systems development. LT acts in a broader context, within the environment that the software operates the company's value flows. Poppendick and Cusumano[6] recommend that organizations should start with some AM like Extreme Programming (XP) or Scrum to experiment with LT results. Such approaches could be considered as a starting point, and would be adapted and improved over time by the people and teams performing the work.

AM can be considered to be a first milestone concerning the influence of LT in SE, that has increasingly influenced the software industry and researchers. Nowadays it is widely used [8]. However, two decades after the introduction of the AM, LT still influences important trends in the SE field, being present in new approaches and paradigms, such as DevOps, Enterprise Agile, Lean Startup, Continuous Integration, Continuous Software Engineering, among others [5][9].
The purpose of this work is to carry out a systematic review with the objective of identifying the practices and approaches proposed by SE researchers in the last years that have an explicit LT influence. This work is part of a broader research that investigates the adoption of LT in software development companies, and aims to contribute to future research, showing the extent to which the efforts of the academic community are focused on this line of research and the opportunities of new works.

This article is structured in the following way: Section 2 describes the revision planning, the parameters adopted to use the search engines and the selection criteria of the researched papers. The leading of systematic reviews and the preliminary selection of these studies are presented in Section 3. Section 4 shows the final selection and the analysis of the studies that entered the Systematic Review. In Section 5 a critical analysis is presented on the practices and approaches of SE that show explicit LT influence. Finally, Section 6 presents the conclusions related to this work.

2. PLANNING

The planning activities of this Systematic Review (SR) were carried out in accordance with the recommendations of the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses), described in Galvão, Pansani and Harrad[10], and with the protocol for researches concerning revisions in the area of software engineering presented by Biolchini et. al. [11]. Planning is the stage at which the research objective should be defined, how SR will be performed and what criteria will be taken into account for inclusion or exclusion of work. This section presents the details of this review step.

2.1. Research Objectives

The purpose of this work is to gather practices and approaches developed by SE researchers in recent years who have worked explicitly under LT influence.

2.2. Formulation of Research Questions: Focus and Specificities

The main focus of research questions is to identify papers that relate SE to LT. The questions were elaborated so that the selected papers have as its central topic the practices and approaches of the SE that are influenced by LT and that report information of its applications in the field.

- Research Question 1 (RQ1): what LT principles and tools are being adapted to the SE domain in current research?
- Research Question 2 (RQ2): what are the SE practices and approaches used by software development teams explicitly influenced by LT principles?
- Research Question 3 (RQ3): in what areas of knowledge of SE are the identified studies commonly applied?

The specificities as to the quality and breadth of the research questions are described below:

- Intervention: SE practices and approaches influenced by LT principles;
- Control: this review began with the papers of [1][2][5][12][13]
- Population: SE practices and approaches applied to software development;
- Results: to identify which SE practices and approaches in recent years have been influenced by LT principles and their respective areas of knowledge;
- Application: This work will provide SE researchers and understanding of the breadth of LT application in the software industry today and future work opportunities.

2.3. Search Strategy for Selections of Studies

Initially, the criteria for selection of the sources were defined and the search methods which would be considered. Thus, an initial set of research sources was identified. After that, the language was defined that would be considered and, thus, the initial set of search sources was reduced. Finally, the keywords and the search string were defined. Keywords were defined according to the purpose and the research questions presented in this review. This process and all its features are described below:

- Source Selection Criteria: Only indexed databases and Internet search engines were selected. In the selected mechanisms, filters were used for the year and the search string;
- Source search methods: a search string was used in databases and search engines. This string was used to filter titles and abstracts of articles;
- Keywords: software engineering, software development and lean;
- Listing of sources: second [14] it is recommended to search in specific search bases and to use at least one general search engine. The general basis consulted was Science Direct and the Scopus journal base aggregator. The specific basis was the electronic search engine IEEE Xplore;
- Types of studies: studies were carried out in articles, papers presented at conferences, patents and reviews;
- Language of studies: the language used was English because of its wide international scientific acceptance.

2.4. Criteria and Procedures for Selection of Studies

The most suitable papers for this SR were selected using inclusion, exclusion and search string constraints, restricting articles published to the years 2012 to 2016. Initially, an evaluation of the title and abstract of each identified work was carried out. Subsequently, the documents that best

fit the inclusion criteria proposed in this article were pre-selected. The inclusion and exclusion criteria were specified according to the search strategies and the questions proposed in this work. The following are the evaluation criteria for the inclusion and exclusion of studies.

2.4.1. Inclusion Criteria

The objective of the inclusion criteria is to qualify the relevance of the work evaluated according to the proposal of this SR. These criteria are described below:

- Inclusion Criteria 1 (I1): the papers must be in digital format and available free of charge on the internet or by means of an agreement with the educational institution where this research was carried out;
- Inclusion Criteria 2 (I2): complete papers written in English;
- Inclusion Criteria 3 (I3): studies related to software engineering or software development;
- Inclusion Criterion 4 (I4): the principles of LT are reported or related in the studies.

2.4.2. Exclusion Criteria

For this systematic review, the purpose of the exclusion criteria is to disregard irrelevant work. These criteria are described below:

- Exclusion Criteria 1 (E1): consider software development as a secondary or auxiliary theme in the selected work;
- Exclusion Criteria 2 (E2): explicitly state the influence of LT principles on some practice or approach proposed and reported in the studies;
- Exclusion Criteria 3 (E3): consider the term "*lean*" and its variations in a different context from the focus of this paper.

2.4.3. Search String

The purpose of the search string was to find studies relating software engineering or software development to LT. Due to variations and combinations related to the "*lean*" concept, such as "*lean thinking*", "*lean production*", "*lean six sigma*", "*lean enterprise*" and "*lean software development*", among others, it was decided to use in the search string a more generic and common word to these variations. The goal was to broaden the research domain. Thus, the search string was defined as: ("*software engineering*" OR "*software development*") AND "*lean*".

2.5. Study Selection Process

During the initial selection process, the search string was used as a search parameter in the indexed sources. After the selection procedure of these articles, a filing procedure was carried out to ensure that each document was considered only once. Subsequently, the studies were shared so that the two reviewers could read the summary and conclusion.

Each reviewer reviewed the work according to the inclusion and exclusion criteria presented in Section 2.4. At the end of the reading it was defined whether or not the work was suitable for this SR. In situations where there were divergences between the analyses of the researchers, discussions took place between them until a consensus was obtained. At the end of the evaluation of each document, each of these reviewers recorded the reason for inclusion or exclusion of the respective article. In the final selection, the selected papers, because they were relevant to SR, were read completely.

Finally, the articles were evaluated according to the research questions presented in Section 2.2 and the data summarized for analysis in this review.

3. IMPLEMENTATION OF THE SR PROTOCOL

Researches were carried out using search engines in digital libraries available in the CAPES portal. First, the search of the IEEE Xplore database was performed. The search string was then applied to the Science Direct journals portal. Finally, this string was searched in the Scopus portal reference aggregator.

Of the 549 papers recovered, 33 belonged to IEEE Xplore, 342 were from Science Direct and 174 were found in Scopus. To organize these studies and to assist in the collaborative work of the reviewers, an online spreadsheet was used. Among all the recovered works, there were 42 duplicate articles. After removing this redundancy, 507 studies remained. After the work selection process, described in Section 2.5, 80 articles were left to be read out for synthesis in this SR. Figure 1 presents a summary of this selection process.

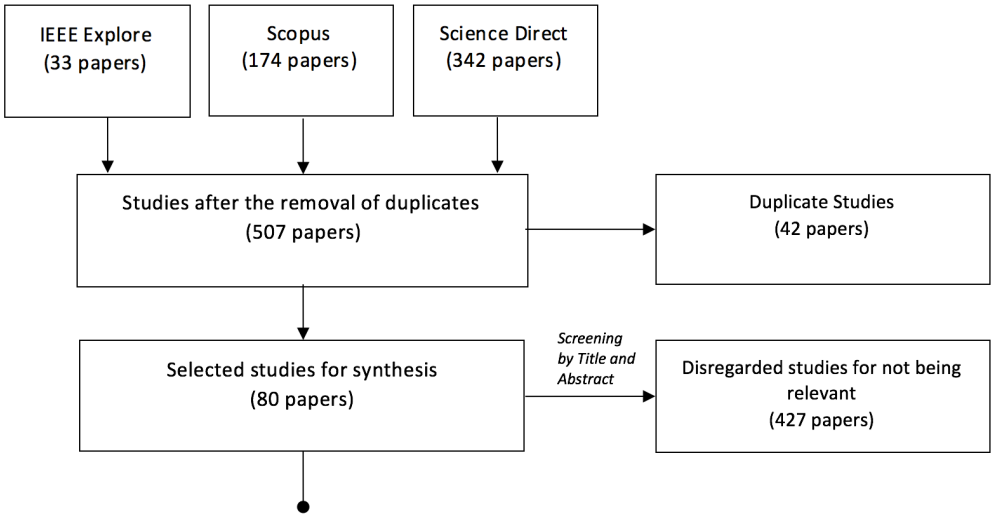

Figure 1. Selection process of the papers

After the complete reading of the papers, the two reviewers prepared a categorization worksheet to classify them according to the relevant aspects proposed in the research questions of this study.

3.1. Rating the studies

In order to respond to RQ1, the studies were ranked based on adaptations of LT principles to the SE domain. Three references identified in the papers selected for SR were selected. Subsequently, they were summarized in Table 1. Some of these principles present overlaps, reflecting the core and essence of LT.

Table 1. Principles of LT relevant to SE

Principles of Lean Software Development (LSD) [15]	Principles for the Development of Products (DP) [16]	Kanban's Principles (KB) [17]
• Eliminate waste • Integrate quality • Create knowledge • Dismiss decisions • Deliver quickly • Respect people • Optimize the whole	• Use an economic vision • Manage rows • Explore variability • Reduce the batch size • Apply WIP restrictions • Use quick feedback • Decentralize control	• View the workflow • Limit WIP • Manage the flow • Make explicit process policies • Collaborative improvement • Decentralize control

The ratings of the RQ1 and RQ2 data were performed as the work was thoroughly read by the reviewers. As response elements were identified, they were also classified and summarized according to the frequencies of the references. In this SR, 27 LT tools adapted to the SE domain were identified and are currently applied by this industry. We also classified 17 approaches and 35 SE practices applied by software development teams explicitly under the influence of LT principles. The RQ3 sought to obtain the distribution of the studies identified in the knowledge areas of SE [18]. Some papers were classified in more than one area of knowledge due to their greater comprehensiveness.

The mappings and syntheses of relevant literature papers for the research questions have been elaborated and are presented in detail in Section 4 of this article.

3.2. Threats to Research Validity and Limitations

The main limitations of the review are biases in the selection of publications and inaccuracy in data extraction. To help ensure that the selection process was unbiased, a research protocol was used that defined the research questions in advance. These questions used keywords and search terms that allowed the identification of relevant literature. However, it is important to recognize that SE keywords are not standardized and may be in a specific language. Therefore, because of the choice of keywords, there is a risk that relevant studies have been omitted.

Another factor that threatens the validity of the research may be present in the subjective decisions that occurred during the selection process and extraction of some studies that did not present a clear description, consequently hindering the objective application of the criteria and analysis, especially in relation to RQ2 question. To minimize this threat, the selection and extraction was done in an iterative and collaborative way by the authors, and possible conflicts in individual interpretations were discussed.

4. RESULTS

The objective of this section is to summarize the information collected in the selected papers for this SR and thus contribute with the answers to the three research questions described in Section 2.2.

In order to respond to RQ1, we sought to establish which LT principles and tools are being adapted to the domain of SE today. The ratings for data on this question were carried out as the work was read and respected the arguments of the authors of the studies. Non-exclusive citations of these adaptations of LT principles were found in Table 2. Most of the 80 papers selected,

51.2% (41 studies), use adaptations of LT principles suggested by LSD [15]. The principles adapted for DP [16] and KB [17] summarized 14.1% and 17.3%, respectively. Some studies had no direct mention of these adaptations.

Table 2. Adaptations of the LT Principles in SE

Origin	# of studies	%	References
Principles of Lean Software Development (LSD)	41	51.2%	[19][20][21][22][23][24][25][26][27][28][29][30][31][32][33][34][35][36][37][38][39][6][40][41][42][43][44][45][46][47][48][49][50][51][52][53][54][55][56][57] [58]
Principles for the Development of Products (DP)	8	10.0%	[19][59][22][30][34][60][36][48]
Kanban's Principles (KB)	13	16.2%	[19][61][62][27][29][63][41][64][65][66][48][49][52]

When analyzing in detail only the works that have quotations based on LSD (41 references), the three most frequent principles were: eliminate waste (33 references, 80.5%), fast delivery (25 references, 61.0%) and optimize the whole (23 references, 56.1%). In most cases, the articles cited more than one LSD principle.

The conceptual definitions and respective original purposes of the various LT tools are found in detail in publications of this domain [3][67]. In this SR a relatively extensive list of these tools that are being applied to the SE domain has been identified. Table 3 presents a summary of the collection of adaptations collected in the studies. Most frequently, Kanban citations (27 references), Value Stream Mapping (10 references) and Limit Work in Progress (9 references), the latter related to a certain extent to Kanban, are mentioned. However, other LT tools have obtained a single citation, such as: GenchiGenbutsu, SMED, Standardish Work, Obeya and Hanedashi, in [30] and Evidence-Based Decision in [34]. Moreover, some, were never found, as is the case of Total Productive Maintenance, used in production maintenance to eliminate losses and reduce stops of continual processes.

Table 3. Adaptations of LT Tools in SE

LT tool	# of studies	References
Andon	3	[19][68][30]
Management Performance	4	[22][69][65][70]
Kaikaku	2	[30][71]
PokaYoke	3	[27][30][35]
Visual Management	2	[23][40]
Toyota Production System	7	[71][36][39][72][52][53][54]
Kaizen	8	[19][62][29][30][6][44][73][51]
Just-In-time	4	[47][74][52][75]
Takttime	3	[76][47][77]
Kanban	27	[19][61][20][62][23][78][79][31][33][63][6][80][41][69][64][65][44][70][66][74][49][73][51][52][54][55][56]
Value Stream Mapping	9	[19][59][62][68][31][33][81][82][51]

Bottleneck Analysis	2	[19][21]
Jidoka	3	[19][68][30]
Root-Cause Analysis	2	[19][20]
Cumulative Flow Diagram	3	[19][20][21]
PDCA	4	[19][23][72][83]
Limit Work In Progress	9	[19][60][63][36][41][64][66][74][57]
Flow unit	2	[31][47]
Minimum Viable Product	4	[84][80][85][86]
Toyota Product Development	5	[71][39][72][70][54]

The next set of data collected in this SR is related to SE practices and approaches used by software development teams explicitly influenced by LT principles. These are essential parts of RQ2's response. The information was grouped,Figure 2 describes the main approaches and Table 4, the SE tools identified in the papers.

The SE approaches that most appear in the studies are related to AM. There are citations for Scrum (32.1%), Extreme Programming (23.5%) and Kanban (32.1%), some even making generic references to AM (24.7%). Lean Development, or simply Lean, is also treated in a generic way in some researches. Despite the relatively high frequency, 24.7%, the authors mention this term without too many specific details or references. In most cases, the quotation is made along with AM. The highlight of these data is that almost half of the studies are related to the LSD approach (43.2%).

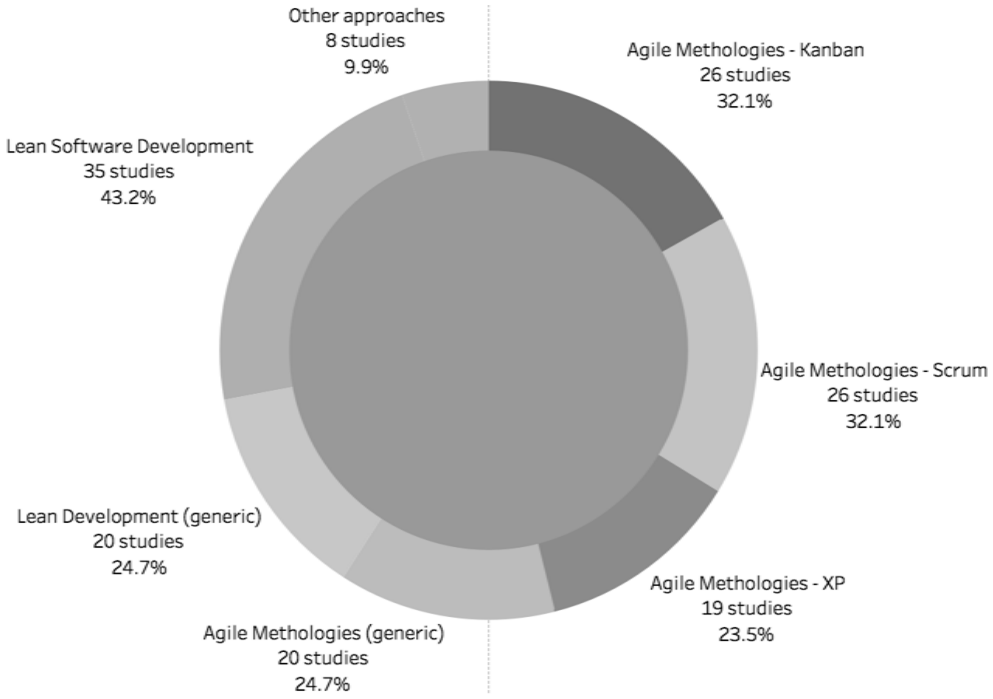

Figure 2. SE Approaches Under LT Influence

Several other SE approaches have been identified. They only add up to 9.9% and have an almost unitarian representation. Quotes were found for: Scaled Agile Framework, Adaptive Development, Rapid Application Development, Large-scale Agile, Feature Driven Development,

Enterprise Agile, Leagile, Scrumban, Incremental Commitment Spiral Model, Dynamic System Development Agile and Crystal Methods.

Likewise, we sought to bring SE practices together under the explicit influence of Lean thinking. 35 practices were identified. The main practices are presented in Table 4. The practices of Continuous Integration (10%), Continuous Deployment (5.0%) and Continuous Delivery (5.0%), defined by Fitzgerald and Stol[5] as subtopics of Continuous Software Engineering (CSE). CSE is presented as a trend for the next few years in the SE area. The authors cite LT as an explicit influence of these three practices. The practice of Test Driven Development is also relevant (6.2%).

Some of the SE practice citations are unique to the respective papers or split with one more study. In addition to the practices presented in Table 4, Agile Modeling, Assessment Lean for Software, Automated tests, Automatic Code Metrics, Behavior-Driven Development, Beyond Budgeting, Coding Dojo, Collective ownership, Continuous Software Engineering, Communities of Practice, Coding Standards, Design Thinking, Discount Usability Engineering, Frequent reviews, Lean Configuration Management, Lean Data Science, Lean UX, Maintenance Metrics, Microservices, Minimum Marketable Feature, Organizational Patterns, Pair Programming, Rapid Contextual Design, Real-Time Value Delivery, Refactoring, Scrum-of-Scrum, Simple Design, Dojo Testing, Test-Driven Design and Unit Test. It should be highlighted that 69.1% of the papers do not explicitly mention any SE practice, having a greater focus on the software development approach.

Table 4. Main SE practices under LT influence

The pratices ofSE	# of studies	%	References
Test Driven Development	5	6.2%	[25][27][47][48][51]
DevOps	4	5.0%	[30][87][84][44]
Continuous Integration	8	10.0%	[23][25][27][30][31] [87][84][39]
Continuous Deployment	4	5.0%	[30][31][84][44]
Continuous Delivery	4	5.0%	[30][31][87][6]

The distribution of the papers in this SR concerning the knowledge areas of SE [18] was summarized in Table 5 below. It is worth noting the concentration of studies in the area of Software Engineering Models and Methods (62.5%). There is an intermediate group of approximately 10% of the studies: Software Engineering Process, Software Engineering Management and Software Quality. The two areas with the lowest number of referenced articles are Software Testing (5.0%) and Software Maintenance (6.2%).

Table 5. Distribution by SE Knowledge Area

Knowledge Area	# of studies	%	References
Software Construction	5	6.2%	[23][79][30][31][88]
Software Configuration Management	8	10.0%	[23][28][30][31][87][42][89][88]
Software Engineering Management	14	17.5%	[61][59][22][62][23][78][28][79][30] [34][90][85][51][58]
Software Maintenance	5	6.2%	[21][79][30][31][32]
Software Engineering Models and Methods	50	62.5%	[19][22][91][25][26][92][93][27][78] [28][79][30][31][94][33][81][84][60] [63][71][36][95][37][38][39][72][6][

			40][69][43][64][65][44][45][70] [66][47][77][96][48][49][73][52][53] [54][83][55][97][56][57]
Software Engineering Professional Practice	6	7.5%	[25][78][30][98][74][53]
Software engineering Process	16	20.0%	[59][23][24][25][68][78][28][79][29] [30][31][94][87][76][50]
Software Design	7	8.7%	[20][23][30][32][35][80][41]
Software Quality	10	12.5%	[23][25][28][79][30][82][86][56]
Software Requirements	6	7.5%	[59][23][78][30][80][46]
Software Testing	4	5.0%	[23][30][31][42]

5. DISCUSSION

In this section, we will analyze the research questions of this work, starting by discussing what was found in this SR about the practices and approaches of the SE explicitly described under the influence of the LT and later the limitations and threats will be presented to this systematic review.

5.1. Which LT principles and tools are being adapted to the ES domain in current research (QP1)?

This SR identified that LT principles adapted to the SE domain by LSD [15] are present more frequently in papers partaining to the area, totaling more than half of the selected total. This is also reflected in a relevant presence of articles (43.2%) that use the SE approach suggested by the same authors.

When considering studies of LT tools used in SE, Kanban is most frequently used, with 1/3 of the citations (27 references) followed by Value Stream Mapping (9 references). Some of the tools identified have only one or two citations of works. For the Total Productive Maintenance tool, no related papers were found. The low frequency of citations in papers may indicate that there is still room for researchers and practitioners to experiment with new applications of these tools in the field of software development.

5.2. What are the SE practices and approaches used by software development teams explicitly influenced by the principles of LT (QP2)?

The approaches of SE influenced by LT principles with higher incidence in this SR are related to AM. Although there are some generic quotes to AM (24.7%), there are specific citations for Scrum (32.1%), Extreme Programming (23.5%) and Kanban (32.1%). Lean Software Development (43.2%) also appears in a significant way.

Although some authors point to a certain conceptual confusion and scope between AM and LSD [19] [68], there are many points in common between them, and both are heavily influenced by LT from its origins. Wang et. al. [19] even point out a proposal for this conflict through Leagile Development, trying to combine the benefits of both approaches. Other authors [99] have observed that there is a shift going from AM to LSD in recent times. LSD, although originally seen as just another AM, is having an increasing focus, currently being viewed as a separate category, rather than as an instance of agile methods. In this SR the quotes that relate to AM and LSD come to sum up an expressive volume of works, with 30 citations (37.5%).

When compared to specific AM references, such as Scrum, Extreme Programming and Kanban, and LSD, the values collected are very similar. As each article could consider one or more approaches, it is difficult to ponder which approach is the most influential one in SE today. A relatively similar distribution of approaches among researchers is noted, with none of them taking prominence in community preference.

The SE practices collected in the papers were summarized in Table 4. Continuous Integration emerged more frequently (10.0%). In a total of 35 rated practices, two relevant points are observed. The first is that there is not a strong concentration on a specific practice. The second point is that 69.1% of the papers do not explicitly mention any SE practice, only the approach used. We can consider several hypotheses for this phenomenon, however, it is not the main objective of this work to make this analysis, and may be the object of a future project.

5.3. In what areas of SE knowledge are these identified studies commonly applied?

The data indicate that the areas of Software Engineering Models and Methods have approximately 2/3 of the studies and are accompanied by an intermediate group, which includes Software Engineering Processes, Software Engineering Management and Software Quality. The two areas with the lowest number of studies referenced are Software Testing and Software Maintenance, which may indicate an opportunity for research by the academic community because these are areas that are still under-explored.

6. CONCLUSIONS

This SR sought in literature, reports of practices and approaches of SE that suffer explicit influence from the principles of LT. For this review, papers published between the years of 2012 and 2016 were selected. Of the 549 publications related to the proposed objective, 80 papers were analyzed in detail to answer the research questions.

The adaptation of the LT principles to the domain of SE most used in the selected studies is Lean Software Development [15]. We classified 17 LT tools adapted for SE. Most of these papers address Kanban and Value Stream Mapping. There are still opportunities for research into new adaptations, since for some LT tools the references are very rare and, in some cases, absent.
The papers also point out that AMs are the majority of the citations in SE approaches under the influence of LT. There is a relevant volume of citations for Scrum, Extreme Programming and Kanban and Lean Software Development methodologies, including, more than 1/3 of them associating AM to LSD. Among SE practices, researches on Continuous Integration, Continuous Deployment, Continuous Delivery, DevOps and Test Driven Development are more frequent. A total of 35 SE practices currently used by the software industry related to LT, were identified.
Studies of this SR were classified according to their respective areas of knowledge of SE. Software Engineering Models and Methods get 2/3 of the studies. The areas of Software Testing and Software Maintenance presented less number of studies and were still little explored.
This systematic review may assist researchers seeking to develop studies in the field of Software Engineering and Lean Thinking, pointing out the current state of academic worksand gaps of opportunity for further research.

REFERENCES

[1] K. B. Stone, "Four decades of lean: a systematic literature review," Int. J. Lean Six Sigma, vol. 3, no. 2, pp. 112–132, 2012.

[2] G. Narayanamurthy and A. Gurumurthy, "Leanness assessment: a literature review," Int. J. Oper. Prod. Manag., vol. 36, no. 10, pp. 1115–1160, 2016.

[3] J. P. Womack, D. T. Jones, and D. Roos, "The Machine that Changed the World: The Story of Lean Production," World. pp. 1–11, 1990.

[4] J. P. Womack and D. T. Jones, Lean Thinking, vol. 5. 1996.

[5] B. Fitzgerald and K. J. Stol, "Continuous software engineering: A roadmap and agenda," J. Syst. Softw., vol. 123, pp. 176–189, 2017.

[6] M. Poppendieck and M. A. Cusumano, "Lean software development: A tutorial," IEEE Softw., vol. 29, no. 5, pp. 26–32, 2012.

[7] S. C. Bell and M. a Orzen, Enabling and Sustaining Your Lean Transformation. 2011.

[8] VersionOne, "10th Annual State of Agile Report," Atlanta, United States, 2015.

[9] G. G. Claps, R. Berntsson Svensson, and A. Aurum, "On the journey to continuous deployment: Technical and social challenges along the way," in Information and Software Technology, 2015, vol. 57, no. 1, pp. 21–31.

[10] D. Moher, A. Liberati, J. Tetzlaff, and D. G. Altman, "Principais itens para relatar Revisões sistemáticas e Meta-análises: A recomendação PRISMA*," Epidemiol. Serv. Saúde, vol. 24, no. 2, pp. 335–342, 2015.

[11] J. C. de Almeida Biolchini, P. G. Mian, A. C. C. Natali, T. U. Conte, and G. H. Travassos, "Scientific research ontology to support systematic review in software engineering," Adv. Eng. Informatics, vol. 21, no. 2, pp. 133–151, 2007.

[12] T. Dingsøyr and T. Dyba, "Empirical studies of agile software development: A systematic review," Inf. Softw. Technol., vol. 50, pp. 833–859, 2008.

[13] E. Papatheocharous and A. S. Andreou, "Empirical evidence and state of practice of software agile teams," in Journal of Software: Evolution and Process, 2014, vol. 26, no. 9, pp. 855–866.

[14] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," Inf. Softw. Technol., vol. 55, no. 12, pp. 2049–2075, 2013.

[15] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit. 2003.

[16] D. G. Reinertsen, "The Principles of Product Development Flow: Second Generation Lean Product Development," Work, vol. 14, p. 304, 2009.

[17] D. J. Anderson, Kanban: Successful Evolutionary Change for Your Technology Business, 3.8.2010. Blue Hole Press, 2010.

[18] IEEE, P. Bourque, and R. E. Fairley, SWEBOK v.3. 2014.

[19] X. Wang, K. Conboy, and O. Cawley, "'Leagile' software development: An experience report analysis of the application of lean approaches in agile software development," J. Syst. Softw., vol. 85, no. 6, pp. 1287–1299, 2012.

[20] K. Power and K. Conboy, "A Metric-Based Approach to Managing Architecture-Related Impediments in Product Development Flow: An Industry Case Study from Cisco," in 2015 IEEE/ACM 2nd International Workshop on Software Architecture and Metrics, 2015, pp. 15–21.

[21] K. Petersen, "A Palette of Lean Indicators to Detect Waste in Software Maintenance: A Case Study," 2012, pp. 108–122.

[22] V. Liubchenko, "A review of agile practices for project management," in 2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT), 2016, pp. 168–170.

[23] K. N. Manjunath, J. Jagadeesh, and M. Yogeesh, "Achieving quality product in a long term software product development in healthcare application using Lean and Agile principles: Software engineering and software development," in 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013, pp. 26–34.

[24] T. Karvonen, P. Rodriguez, P. Kuvaja, K. Mikkonen, and M. Oivo, "Adapting the Lean Enterprise Self-Assessment Tool for the Software Development Domain," in 2012 38th Euromicro Conference on Software Engineering and Advanced Applications, 2012, pp. 266–273.

[25] T. Scheerer, Alexander; Schmidt, Christoph T.; Heinzl, Armin; Hildenbrand and D. Voelz, "Agile Software Engineering Techniques: The Missing Link in Large Scale Lean Product Development," in Lecture Notes in Informatics - Proceedings, 2013, pp. 319–330.

[26] B. Boehm, J. Lane, and S. Koolmanojwong, "An Orthogonal Framework for Improving Life Cycle Affordability," Procedia Comput. Sci., vol. 16, pp. 1170–1179, 2013.

[27] P. Rodríguez, K. Mikkonen, P. Kuvaja, M. Oivo, and J. Garbajosa, "Building lean thinking in a telecom software development organization: strengths and challenges," in Proceedings of the 2013 International Conference on Software and System Process - ICSSP 2013, 2013, p. 98.

[28] U. K. Durrani, Z. Pita, and J. Richardson, "Coexistence of agile and SCM practices," J. Syst. Inf. Technol., vol. 16, no. 1, pp. 20–39, Mar. 2014.

[29] M. Paasivaara and C. Lassenius, "Communities of practice in a large distributed agile software development organization – Case Ericsson," Inf. Softw. Technol., vol. 56, no. 12, pp. 1556–1577, Dec. 2014.

[30] B. Fitzgerald and K.-J. Stol, "Continuous software engineering and beyond: trends and challenges," in Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering - RCoSE 2014, 2014, pp. 1–9.

[31] T. Lehtonen, T. Kilamo, S. Suonsyrja, and T. Mikkonen, "Continuous, Lean, and Wasteless: Minimizing Lead Time from Development Done to Production Use," in 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2016, pp. 73–77.

[32] J. Werewka, K. Jamróz, and D. Pitulej, "Developing Lean Architecture Governance at a Software Developing Company Applying ArchiMate Motivation and Business Layers," 2014, pp. 492–503.

[33] N. Bin Ali, K. Petersen, and B. B. N. de França, "Evaluation of simulation-assisted value stream mapping for software product development: Two industrial cases," Inf. Softw. Technol., vol. 68, pp. 45–61, Dec. 2015.

[34] B. Fitzgerald, M. Musiał, and K.-J. Stol, "Evidence-based decision making in lean software project management," in Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014, 2014, pp. 93–102.

[35] T. Hayata, J. Han, and M. Beheshti, "Facilitating Agile Software Development with Lean Architecture in the DCI Paradigm," in 2012 Ninth International Conference on Information Technology - New Generations, 2012, pp. 343–348.

[36] K. Power and K. Conboy, "Impediments to Flow: Rethinking the Lean Concept of 'Waste' in Modern Software Development," 2014, pp. 203–217.

[37] A. Shalloway, "Integrating lean thinking to achieve multi-product and multi-team agility," Cut. IT J., vol. 26, no. 6, pp. 18–23, 2013.

[38] M. Shcherbakov, N. Shcherbakova, A. Brebels, T. Janovsky, and V. Kamaev, "Lean Data Science Research Life Cycle: A Concept for Data Analysis Software Development," in Communications in Computer and Information Science (CCIS), 2014, pp. 708–716.

[39] M. Misaghi and I. Bosnic, "Lean Mindset in Software Engineering: A Case Study in a Software House in Brazilian State of Santa Catarina," in Communications in Computer and Information Science (CCIS), Springer, Cham, 2014, pp. 697–707.

[40] U. Viswanath, "Lean Transformation: Adapting to the change, factors for success and lessons learnt during the journey: A case study in a multi location software product development team," in Proceedings of the 9th India Software Engineering Conference on - ISEC '16, 2016, pp. 156–162.

[41] R. L. Nord, I. Ozkaya, and R. S. Sangwan, "Making Architecture Visible to Improve Flow Management in Lean Software Development," IEEE Softw., vol. 29, no. 5, pp. 33–39, Sep. 2012.

[42] M. V. Mäntylä, B. Adams, F. Khomh, E. Engström, and K. Petersen, "On rapid releases and software testing: a case study and a semi-systematic literature review," Empir. Softw. Eng., vol. 20, no. 5, pp. 1384–1425, Oct. 2015.

[43] B. Boehm, S. Koolmanojwong, J. A. Lane, and R. Turner, "Principles for Successful Systems Engineering," Procedia Comput. Sci., vol. 8, pp. 297–302, 2012.

[44] L. E. Lwakatare, P. Kuvaja, and M. Oivo, "Relationship of DevOps to Agile, Lean and Continuous Deployment," in Lecture Notes in Computer Science (LNCS), 2016, pp. 399–415.

[45] M. Staron, W. Meding, and K. Palm, "Release Readiness Indicator for Mature Agile and Lean Software Development Projects," in Lecture Notes in Business Information Processing (LNBIP), 2012, pp. 93–107.

[46] Z. Z. R., Salo; Poranen T., "Requirements management in GitHub with a lean approach," in 14th Symposium on Programming Languages and Software Tools (SPLST'15), 2015, pp. 164–178.

[47] B. S. Blau, T. Hildenbrand, R. Knapper, A. Mazarakis, Y. Xu, and M. G. Fassunge, "Steering through Incentives in Large-Scale Lean Software Development," in Communications in Computer and Information Science, Vol 275., Berlin: Springer, 2013, pp. 32–48.

[48] P. Rodríguez, J. Markkula, M. Oivo, and K. Turula, "Survey on agile and lean usage in finnish software industry," in Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '12, 2012, p. 139.

[49] H. Jonsson, S. Larsson, and S. Punnekkat, "Synthesizing a Comprehensive Framework for Lean Software Development," in 2013 39th Euromicro Conference on Software Engineering and Advanced Applications, 2013, pp. 1–8.

[50] K. D. Palmer, "The Essential Nature of Product Traceability and its Relation to Agile Approaches," Procedia Comput. Sci., vol. 28, pp. 44–53, 2014.

[51] I. Nurdiani, J. Barstler, and S. A. Fricker, "The impacts of agile and lean practices on project constraints: A tertiary study," J. Syst. Softw., vol. 119, pp. 162–183, 2016.

[52] O. Al-Baik and J. Miller, "The kanban approach, between agility and leanness: a systematic review," Empir. Softw. Eng., vol. 20, no. 6, pp. 1861–1897, Dec. 2015.

[53] Yilang Wu, K. Sato, Lei Jing, Junbo Wang, and Zixue Cheng, "The lean awareness in software-intensive engineering: Experience from one project," in 2015 IEEE 7th International Conference on Awareness Science and Technology (iCAST), 2015, pp. 168–173.

[54] J. Pernstål, R. Feldt, and T. Gorschek, "The lean gap: A review of lean approaches to large-scale software systems development," J. Syst. Softw., vol. 86, no. 11, pp. 2797–2821, Nov. 2013.

[55] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen, "Towards Rapid Releases in Large-Scale XaaS Development at Ericsson: A Case Study," in 2014 IEEE 9th International Conference on Global Software Engineering, 2014, pp. 16–25.

[56] E. Kupiainen, M. V. Mäntylä, and J. Itkonen, "Using metrics in Agile and Lean Software Development – A systematic literature review of industrial studies," Inf. Softw. Technol., vol. 62, pp. 143–163, Jun. 2015.

[57] N. Behroozi and A. Kamandi, "Waste elimination of agile methodologies in web engineering," in 2016 Second International Conference on Web Research (ICWR), 2016, pp. 102–107.

[58] Jaroslav Skrabálek; Christina Böhm, "Why modern mobile and web-based development need a lean agile web approach (LAWA)," in 21st Interdisciplinary Information Management Talks, 2013, pp. 225–232.

[59] J. Heidenberg, M. Weijola, K. Mikkonen, and I. Porres, "A Model for Business Value in Large-Scale Agile and Lean Software Development," 2012, pp. 49–60.

[60] M. Walter, R. Tramontini, R. M. Fontana, S. Reinehr, and A. Malucelli, "From Sprints to Lean Flow: Management Strategies for Agile Improvement," 2015, pp. 310–318.

[61] R. Turner, "A lean approach to scheduling systems engineering resources," CrossTalk, pp. 1–7, 2013.

[62] E. and F. E. P. Corona, "A Review of Lean-Kanban Approaches in the Software Development," WSEAS Trans. Inf. Sci. Appl., vol. 10, no. 1, pp. 1–13, 2013.

[63] R. Turner and J. A. Lane, "Goal-question-Kanban: Applying Lean Concepts to Coordinate Multi-level Systems Engineering in Large Enterprises," Procedia Comput. Sci., vol. 16, pp. 512–521, 2013.

[64] D. I. K. Sjoberg, A. Johnsen, and J. Solberg, "Quantifying the Effect of Using Kanban versus Scrum: A Case Study," IEEE Softw., vol. 29, no. 5, pp. 47–53, Sep. 2012.

[65] M. Olszewska (née Pląska), J. Heidenberg, M. Weijola, K. Mikkonen, and I. Porres, "Quantitatively measuring a large-scale agile transformation," J. Syst. Softw., vol. 117, pp. 258–273, Jul. 2016.

[66] A. Tregubov and J. A. Lane, "Simulation of Kanban-based Scheduling for Systems of Systems: Initial Results," Procedia Comput. Sci., vol. 44, pp. 224–233, 2015.

[67] J. Liker, The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer. New York, NY, 2004.

[68] G. Anand, A. Chandrashekar, and G. Narayanamurthy, "Business Process Reengineering Through Lean Thinking: A Case Study," J. Enterp. Transform., vol. 4, pp. 123–150, 2014.

[69] F. Fagerholm, M. Ikonen, P. Kettunen, J. Münch, V. Roto, and P. Abrahamsson, "Performance Alignment Work: How software developers experience the continuous adaptation of team performance in Lean and Agile environments," Inf. Softw. Technol., vol. 64, pp. 132–147, Aug. 2015.

[70] P. Kettunen and S. Moilanen, "Sensing High-Performing Software Teams: Proposal of an Instrument for Self-monitoring," in Lecture Notes in Business Information Processing (LNBIP), Vol 111., Berlin, Heidelberg: Springer, 2012, pp. 77–92.

[71] P. J. Lane, Michael; Fitzgerald, Brian; Agerfalk, "Identifying lean software development values," in Proceedings of 21st European Conference on Information Systems (ECIS), 2012, pp. 1–12.

[72] F. T. Shah, S. Shamail, and N. Ahmad Akhtar, "Lean quality improvement model for quality practices in software industry in Pakistan," J. Softw. Evol. Process, vol. 27, no. 4, pp. 237–254, Apr. 2015.

[73] R. J. Kusters, F. M. Munneke, and J. J. M. Trienekens, "The Impact of Lean Techniques on Factors Influencing Defect Injection in Software Development," in Proceedings of the 17th International Conference on Enterprise Information Systems, 2015, pp. 412–419.

[74] M. O. Ahmad, K. Liukkunen, and J. Markkula, "Student perceptions and attitudes towards the software factory as a learning environment," in 2014 IEEE Global Engineering Education Conference (EDUCON), 2014, pp. 422–428.

[75] D. Taibi, A. Janes, and V. Lenarduzzi, "Towards a Lean Approach to Reduce Code Smells Injection: An Empirical Study," in Lecture Notes in Business Information Processing, Vol 251., Springer, 2016, pp. 300–304.

[76] J. Choudhury and B. Thushara, "Software Documentation in a Globally Distributed Environment," in 2014 IEEE 9th International Conference on Global Software Engineering, 2014, pp. 90–94.

[77] U. Samanta and V. S. Mani, "Successfully Transforming to Lean by Changing the Mindset in a Global Product Development Team," in 2015 IEEE 10th International Conference on Global Software Engineering, 2015, pp. 135–139.

[78] K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: A systematic literature review," J. Syst. Softw., vol. 119, pp. 87–108, Sep. 2016.

[79] P. R. I. P. A. M. D. Jankovic, "Combining Agile and Traditional Methodologies in Medical Information Systems Development Process," in 5th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Applications, 2016, pp. 65–72.

[80] L. A. Liikkanen, H. Kilpiö, L. Svan, and M. Hiltunen, "Lean UX: the next generation of user-centered agile development?," in Proceedings of the 8th Nordic Conference on Human-Computer Interaction Fun, Fast, Foundational - NordiCHI '14, 2014, pp. 1095–1100.

[81] N. Bin Ali, K. Petersen, and K. Schneider, "FLOW-assisted value stream mapping in the early phases of large-scale software development," J. Syst. Softw., vol. 111, pp. 213–227, Jan. 2016.

[82] M. Staron, W. Meding, and M. Caiman, "Improving Completeness of Measurement Systems for Monitoring Software Development Workflows," in Lecture Notes in Business Information Processing (LNBIP), 2013, pp. 230–243.

[83] P.-W. Ng, "Theory based software engineering with the SEMAT kernel: preliminary investigation and experiences," in Proceedings of the 3rd SEMAT Workshop on General Theories of Software Engineering - GTSE 2014, 2014, pp. 13–20.

[84] J. Järvinen, T. Huomo, T. Mikkonen, and P. Tyrväinen, "From Agile Software Development to Mercury Business," 2014, pp. 58–71.

[85] B. H. Ximenes, I. N. Alves, and C. C. Araújo, "Software Project Management Combining Agile, Lean Startup and Design Thinking," in Lecture Notes in Computer Science, Vol 9186., Springer, 2015, pp. 356–367.

[86] J. Yli-Huumo, T. Rissanen, A. Maglyas, K. Smolander, and L.-M. Sainio, "The Relationship Between Business Model Experimentation and Technical Debt," in Lecture Notes in Business Information Processing, Vol 210., Springer, 2015, pp. 17–29.

[87] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "DevOps," IEEE Softw., vol. 33, no. 3, pp. 94–100, May 2016.

[88] M. Isomursu, A. Sirotkin, P. Voltti, and M. Halonen, "User Experience Design Goes Agile in Lean Transformation -- A Case Study," in 2012 Agile Conference, 2012, pp. 1–10.

[89] N. Agarwal, R. Karimpour, and G. Ruhe, "Theme-Based Product Release Planning: An Analytical Approach," in 2014 47th Hawaii International Conference on System Sciences, 2014, pp. 4739–4748.

[90] A. Maglyas, U. Nikula, and K. Smolander, "Lean Solutions to Software Product Management Problems," IEEE Softw., vol. 29, no. 5, pp. 40–46, Sep. 2012.

[91] V. Garousi, A. Coşkunçay, A. Betin-Can, and O. Demirörs, "A survey of software engineering practices in Turkey," J. Syst. Softw., vol. 108, pp. 148–177, Oct. 2015.

[92] P. Rodríguez, J. Markkula, M. Oivo, and J. Garbajosa, "Analyzing the Drivers of the Combination of Lean and Agile in Software Development Companies," 2012, pp. 145–159.

[93] T. F. Vranić, "Animating organizational patterns," in ighth International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE '15), 2015, pp. 8–14.

[94] M. Paasivaara and C. Lassenius, "Deepening Our Understanding of Communities of Practice in Large-Scale Agile Development," in 2014 Agile Conference, 2014, pp. 37–40.

[95] B. Swaminathan and K. Jain, "Implementing the Lean Concepts of Continuous Improvement and Flow on an Agile Software Development Project: An Industrial Case Study," in 2012 Agile India, 2012, pp. 10–19.

[96] M. Paasivaara, O. Väättänen, M. Hallikainen, and C. Lassenius, "Supporting a Large-Scale Lean and Agile Transformation by Defining Common Values," in Lecture Notes in Business Information Processing, Vol 199., Springer, 2014, pp. 73–82.

[97] S. Dragicevic, S. Celar, and L. Novak, "Use of Method for Elicitation, Documentation, and Validation of Software User Requirements (MEDoV) in Agile Software Development Projects," in 2014 Sixth International Conference on Computational Intelligence, Communication Systems and Networks, 2014, pp. 65–70.

[98] F. Fagerholm and M. Pagels, "Examining the Structure of Lean and Agile Values among Software Developers," 2014, pp. 218–233.

[99] L. X. Wang, M. Lane, and K. Conboy, "From Agile to Lean: The Perspective of The Two Agile Online Communities of Interest," in 19th European Conference on Information Systems, 2011, pp. 1–7.

Authors

Fernando Sambinelli holds a bachelor's degree in Computer Science from the State University of Maringá, a specialist in Software Process Improvement at the Federal University of Lavras and a Master's Degree in Computer Science from the Methodist University of Piracicaba. He is a professor at the Federal Institute of São Paulo and a PhD student in Technology from the State University of Campinas. His research and market expertise focuses on the area of software process improvement, software engineering, project management, agile methods and Lean IT.

Marcos Augusto Francisco Borges holds a bachelor's degree in Computer Engineering from the State University of Campinas, a master's and a doctorate in Computer Science from the State University of Campinas and a postdoctoral degree from the University of Porto, Portugal, in 2009. He is a professor at the School of Technology in the State University of Campinas and a partner in AuctusQualidade e Gestão. His research focuses on project management, Lean IT, serious games and computational thinking.