dos Santos *et al. Journal of Software Engineering Research and Development* (2018) 6:13
https://doi.org/10.1186/s40411-018-0057-1

Journal of Software Engineering Research and Development

RESEARCH Open Access

On the benefits and challenges of using kanban in software engineering: a structured synthesis study

Paulo Sérgio Medeiros dos Santos*, Alessandro Caetano Beltrão, Bruno Pedraça de Souza and Guilherme Horta Travassos

* Correspondence:
pasemes@cos.ufrj.br
Program of Systems Engineering and Computer Science – PESC/COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil

Abstract

Context: Kanban is increasingly being used in diverse software organizations. There is extensive research regarding its benefits and challenges in Software Engineering, reported in both primary and secondary studies. However, these results have not been synthesized yet.

Goal: to investigate and identify the actual benefits and challenges of using Kanban in Software Engineering to support practitioners in understanding and analyzing the benefits and challenges of adopting Kanban in their software projects.

Method: to use the Structured Synthesis Method to aggregate existing empirically-grounded evidence in the published primary studies regarding using Kanban in Software Engineering.

Results: from the 20 selected primary studies in which over 16 benefits were identified, four had the most robust results in the aggregation, i.e., with the most confidence associated, namely: 'work visibility,' 'control of project activities and tasks,' 'flow of work,' and 'time-to-market.' Furthermore, the 'organizational culture' was identified as the most dominant challenge in Kanban implementations.

Conclusions: Syntheses studies represent a fundamental step in organizing the body of evidence as an empirically-grounded reference for decision-making in practice. The benefits with most confidence indeed appear to be the ones intrinsically linked to the Lean thinking and the Kanban approach. As Kanban originated in the manufacturing, it is interesting to observe this kind of confirmation in the software domain. Still, there are several benefits and challenges which still lacks the appropriate level of evidence. We also noticed the absence of negative results reported in the technical literature. These aspects need the additional attention of the research community.

Keywords: Kanban, Benefits and challenges, Structured synthesis method

1 Introduction

The emphasis on delivering business value was one of the leading driving forces behind the adoption of most agile software development approaches. This goal also motivated the introduction of lean thinking in the software development processes, with the elimination of waste as a core principle – along with the continuous learning through short cycles and frequent builds, and the promotion of late changes and fast iterations (Poppendieck and Cusumano 2012).

These movements took place in the context of a business shift to the digital transformation era, in which disruptive business processes and models are seen as necessary paths to promote competitiveness, mainly for those companies that are not willing to give significant control of their processes to big software vendors (Andriole 2017). Together with the technological evolution (e.g., cloud computing), this compelled the software industry to recall the programming-in-the-small (DeRemer and Kron 1976) principles and to revisiting the overwhelming technical complexity and inflexibility of huge, standardized software systems and processes (Andriole 2017). Hence, the need to eliminating waste with unnecessary complexity or promote late changes to keep the software systems up-to-date with the business processes changes.

Despite its origins in manufacturing, lean principles are continuously being explored in new industries, even among those involving intensive knowledge work as in the case of Software Engineering (SE). Staats et al. (2011) state that "knowledge work not only has a context separate from manufacturing but also differs fundamentally in structure, calling into question lean principles' universal applicability." By one hand, agile software development approaches succeeded attaining goals such as adaptability and iterative processes (Abrantes and Travassos 2013), although being very developer-centric and relatively opaque to management regarding effort estimation, duration, and development costs (Maglyas et al. 2012; Fitzgerald et al. 2014). Lean approaches, on the other hand, are more geared towards quantitative measurement and decision making based on evidence (Fitzgerald et al. 2014).

One of the leading lean approaches used in SE is Kanban, which has been increasingly adopted by software organizations (Versionone 2017). Given its rising in popularity, researchers are increasing their attention to this theme, as can be seen in the four secondary studies analyzing different perspectives regarding Kanban (Corona and Pani 2013; Ahmad et al. 2013; Al-Baik and Miller 2015; Ahmad et al. 2018) covering over 20 primary studies. The technical literature is quite comprehensive reporting evidence regarding the benefits expected from Kanban and the challenges involved in its utilization.

However, despite the essential efforts in organizing a body of knowledge as observed in these four secondary studies (systematic reviews and mappings), there is still a lack of synthesis of the benefits and challenges of Kanban. Research syntheses are essential to provide a summarization, integration, combination, and comparison of findings from different studies. They are proposed on the premise that single studies are limited in the extent to which they may be generalized (Cruzes and Dybå 2011). Thus, a research synthesis represents a vital knowledge tool employed to manage and put scientific findings to use (Santos and Travassos 2016).

The primary goal of this paper is to investigate and identify the benefits and challenges of using Kanban in SE evidenced in the technical literature. It is a fundamental step in organizing an empirically-grounded reference for supporting the decision-making on this subject in SE. Also, the aggregated evidence presented in this paper aims to help software practitioners to understand and analyze the benefits and challenges of adopting and using Kanban in their software projects. Besides, to support SE researchers to identify areas where further research is needed to consolidate, understand or evolve the current knowledge regarding the use of Kanban in SE.

This paper is organized as follows. The next section briefly presents the basic concepts of Kanban and lean thinking. In Section 3, the study methodology is detailed, showing how primary studies were selected and aggregated using the Structured Synthesis Method (SSM). Section 4 describes how the primary studies were analyzed before aggregation. In the SSM, the primary studies have to be translated into diagrammatic representations used to aggregate the studies' outcomes. In Section 5, the aggregation process and results are presented. The main benefits and challenges are explained and detailed. Then, Section 6 examines the results in the light of the existing body of evidence and discusses what can be learned from the aggregation. The threats to the validity of this synthesis study are explored in Section 7, and Section 8 concludes this paper.

2 Background

The Lean Methodology, also known as the Toyota Production System, is a production management process developed by Taiichi Ohno in the 40's in the Japanese manufacturing industry context. In its conception, the term "Lean" was primarily associated with the reduction of costs (Ohno and Bodek 1988) through the "elimination of waste" or "doing more with less" (Conboy 2009). In the course of time, it became also focused on *value* for the customer and flow of work. It is common to refer to the lean concept as "lean thinking," meaning that it is a mental model of how the world works (Poppendieck and Poppendieck 2013). In Womack and Jones (1997) the core lean principles are defined as follows:

- ***Value***: can only be defined by the customer. Also, it must be expressed regarding a specific product;
- ***Value* stream**: the course of action through which a specific product must go to make it available;
- **Flow**: the pursuit of continuous production keeping interruptions at a minimal level;
- **Pull**: the customer pulls the product from the producer when it is needed rather than pushing the products, often unwanted, onto the customer;
- **Perfection**: a virtuous cycle is created by the interaction of the previous four principles. "Organizations begin to accurately specify *value*, identify the entire *value* stream, make the *value*-creating steps for specific products flow continuously, and let customers pull *value* from the enterprise."

The Lean philosophy uses a number of tools to support management its operation. One of these tools is called kanban (based on Toyota Production System). In contrast, there is an adaptation of the kanban, made by Anderson (2010), which is called Capital K (or Kanban). In this paper, our focus is on the latter, the Kanban with Capital K used in the software development context. Kanban is an approach to visualize the workflow of a production system. It makes use of the queue theory to control and improve the *value* stream by aiming attention at the production flow. In SE, David Anderson was the first to use Kanban in 2004 with a software development team at Microsoft. According to Anderson (2010), Kanban has five principles:

- **Visualize workflow**: The board is the primary tool used to visualize and coordinate teamwork. Its columns show a sequence of activities, where the cards represent the features under work;
- **Limit work in progress**: WIP is a way to manage and limit the amount of working in progress. There should always be a way to limit and signal to pull a new task;
- **Measure and manage flow**: different statistics and diagrams can be used to monitor the Kanban process such as cycle/lead time, queue size, and cumulative flow diagrams;
- **Make process policies explicit**: policies are an essential part of assuring that the flow is achieved. They establish the conditions to make the pull system work. They include, for instance, how to assign tasks and activities to developers and when a work item can be pulled from one state to another;
- **Use models to recognize improvement opportunities:** three models are suggested (i) the Theory of Constraints, (ii) a subset of ideas from Lean Thinking that identifies wasteful activities as economic costs, and (iii) some variants that focus on understanding and reducing variability.

In SE, a Kanban system is usually implemented as a board on a wall with columns representing the different development process stages, i.e., the value stream (Poppendieck and Cusumano 2012). Cards are used to describe pieces of work or tasks, which are moved through the chart columns. A typical configuration used in a Kanban chart in the software context contains at least columns for the stages of specification, development, test, and deploy (Corona and Pani 2013). For each column, a limit for the work in progress is determined. As a result, flow and bottlenecks are usually the main issues addressed in daily meetings and play a crucial role in identifying improvement opportunities. Furthermore, as a visual tool, the chart stimulates the value stream evaluation materialized in it, also prompting not only the process improvement itself but also the defined policies supporting it.

3 Study method

The fundamental research question regarding our work refer to *"What are the trends observed in empirical studies available in the technical literature regarding the benefits and challenges of using Kanban in software organizations?"* To answer this research question, we aggregated the results of primary studies regarding Kanban using the SSM. The SSM allows the aggregation of qualitative and quantitative evidence through the use of diagrammatic models (Santos and Travassos 2013). As both qualitative and quantitative research synthesis method, the SSM briefly depicts the essential contextual aspects and informs the effects trend (e.g., positive or negative), as well as a certainty estimation about them. Therefore, the SSM provides balanced information regarding the phenomena, neither aggregating precise quantitative findings nor rich qualitative descriptions.

This blend of integrative and interpretive synthesis (Cruzes and Dybå 2011) was the primary reason for deciding to use the SSM as our research method since the primary studies regarding the use of Kanban in SE report both quantitative and qualitative evidence. In the SSM, interpretative synthesis aspects are concerned with the organization and development of concepts to describe contextual aspects of evidence whereas integrative features are focused on pooling data about cause-effect or moderation relations taking into account the uncertainty estimated for each evidence. Besides, the SSM offers tool support to model and synthesize evidence (Santos et al. 2015; Santos and Travassos 2017a), including facilities for graphical modeling, evidence search, and support for the synthesis. Another essential functionality is the evidence model comparison used to aggregate evidence, which has mechanisms for 'conflict resolution' between the models. The Evidence Factory tool including all the results of the synthesis presented in this paper can be accessed at http://evidencefactory.lens-ese.cos.ufrj.br/synthesis/editor/80416.

In general terms, an SSM synthesis study follows three steps: (i) the selection of primary studies, (ii) the analysis and representation of evidence acquired by such studies, and (iii) evidence synthesis. The basic idea involving these three steps is to collect evidence then represent them from the same perspective so that the results can be consolidated and synthesized. It is similar to statistical meta-analysis studies – which is a kind of integrative synthesis – where the effect size is used to get a uniformed view over the studies outcomes, which is also used in their aggregation (Borenstein et al. 2009).

Next, we describe how we applied the SSM to synthesize the research on benefits and challenges of using Kanban in SE. We also provide some descriptions of the SSM definition and utilization necessary for understanding how this synthesis study was conducted. We refer the reader to the following work Santos and Travassos (2013) for further details regarding the method, and to Martinez-Fernandez et al. (2015), Chapetta (2016), and Santos and Travassos (2017b) to find examples of its application.

3.1 SSM step 1: Selecting primary studies

As there are four secondary studies regarding Kanban including one (Ahmad et al. 2018) that has been recently published, there is no reason to perform some of the typical procedures involved in this step, such as defining a search string and selecting the studies based on inclusion and exclusion criteria. Instead, we used the datasets from these secondary studies to form the set of primary studies to be aggregated. We have taken the primary studies from the two most recent secondary studies (Al-Baik and Miller 2015; Ahmad et al. 2018). Regarding the other two, one (Ahmad et al. 2013) is updated by Ahmad et al. (2018), and the other (Corona and Pani 2013) is focused on the tools available for Kanban boards in software development. Only the papers reporting results from primary studies (i.e., case study, survey, controlled experiment, or simulation study) on using Kanban in SE were selected from the two secondary studies. Grey literature and experience reports that were considered in these two secondary studies were excluded from the synthesis.

Ahmad et al. (2018) enumerate 23 technical papers as primary studies (and other 23 as experience reports). However, we have found that three of them (Corona and Pani 2013; Ahmad et al. 2013; Al-Baik and Miller 2015) were, in fact, secondary studies.

Also, one additional paper was excluded (Heikkilä et al. 2016), since the only challenge it reported, called "Setting up and maintaining Kanban", has not been translated as it did not represent a moderator – more details about the benefits and challenges of using Kanban in SE considered in each primary study is shown in Table 1 (Section 4). Thus, from the 23 primary studies, 19 were included in the synthesis.

Al-Baik and Miller (2015) enumerate 37 papers as studies from which six studies we could classify as primary studies. Only one of the six primary studies was not included in Ahmad et al. (2018) and, thus, was also included in the synthesis. The remaining 31 papers were excluded because of the following reasons: (i) grey literature (19 papers); (ii) experience reports (six papers – all included in Ahmad et al. (2018)); (iii) it is not an empirical study (five papers); and (iv) not reporting results describing the benefits and challenges of Kanban use (one paper). Appendix B lists all the primary studies. It also should be noticed that all included papers are from the Software Engineering realm. That is, they investigate Kanban as a software technology (i.e., a set of techniques and tools employed in software development).

3.2 SSM step 2: Analysis and evidence representation

In this step, the goal is to put the selected primary studies under the same perspective so they can be aggregated. The idea is similar to the statistical meta-analysis, in which all primary studies are represented by a numerical value called effect size and then aggregated by combining those values (Borenstein et al. 2009). In the case of SSM, each primary study is represented by an evidence model, which is denominated theoretical structure. The evidence models describe the primary studies' contextual aspects and the effects/moderators expected from the object of study – Kanban, in this synthesis. These descriptions are used as input for determining the evidence compatibility and for the aggregation itself.

As 20 primary studies were selected, we have had to create 20 theoretical structures. The name theoretical structure is related to the origin of the model constructs, which were taken from a representation created for theory building (Sjøberg et al. 2008). Since the model was adapted for the purpose of research synthesis, we use the name theoretical structure to bring attention to the model structure instead of the epistemological aspects related to theory building. In fact, this emphasis is also reflected in the method name *Structured* Synthesis Method. In the following paragraphs, we describe the evidence model constructs.

The ten semantic constructs used in the theoretical structures are shown in Fig. 2. There are three possible types of *structural* relationships in the representation: *is a*, *part of* and *property of*. All of them have counterparts in UML, respectively: generalization, composition and class attributes. The *is a* and *part of* relationships use the same UML notation for generalization and composition. Dashed connections denote *properties*. The relationships are used to link two types of concepts – *value* and *variable*.

A *value* concept represents a particular variable value, usually an independent variable. Rectangles represent *value* concepts. They are classified in *archetypes* (the root of each hierarchy), *causes* (indicated by the use of bold font and a 'C1' following the name denoting that it is the 'cause 1' (e.g., 'Kanban'), and *contextual aspects* (e.g., 'Distributed

Project'). The four archetypes – activity, actor, system, and technology – were suggested by Sjøberg et al. (2008) in an attempt to capture the typical scenario in SE described by an actor applying a technology to perform activities in a software system.

A *variable* concept focuses on value variations usually associated with a dependent variable. *Variable* concepts are represented by ellipses or parallelograms symbolizing *effects* (e.g., 'Work Visibility') and *moderators* (e.g., 'Training), respectively. Also, *effects* are not connected to *cause* using lines as they are assumed to exist when reading the diagram. Lines are also lacking in the link between *moderators* and the (moderated) *effects*. In this case, a textual hint (*e.g.*, 'M1') is shown beside both the moderated effect and moderator. Both relationships, *cause-effect,* and *moderation*, are denominated *influence* relationships.

A seven-point *Likert* scale is used to indicate an effect size. The scale ranges from strongly negative to strongly positive. It is indicated above the ellipse (e.g., ︽ indicates that 'Collaboration' is between weakly positively and positively affected by 'Kanban' – the number of arrows indicates the value in the scale; ︾ represents strongly negative and ︽ strongly positive, and half arrows indicate a range such as in the case of 'Collaboration'). The other type of *variable* concepts, namely *moderators*, indicates that some positive or negative effect is moderated (i.e., reduced) when it increases or decreases. It has a scale with three values indicating the moderation direction: inversely proportional, indifferent, and directly proportional. For instance, the *moderator* 'Training' has an inversely proportional influence on 'Collaboration,' which means that the more it is present, the less it exerts a moderation influence. The last aspect related to *variable* concepts is the association of a *belief value* (ranging from 0% to 100% or 0 to 1) to estimate the confidence in the observed *effects* and *moderations*. The bar under each element represents the *belief value*, e.g., 'Flow of work' has 47% of *belief value*.

3.2.1 Extracting information to build evidence models

In order to create the evidence models, it is necessary to extract information from the primary studies. The goal is to determine and define the concepts (*contextual aspects*, *moderators*, and *effects*) that will form the evidence model, and to estimate the confidence (i.e., *belief value*) over the variable concepts (*moderators* and *effects*). This is usually performed in two stages one for determining the concepts and other for estimating the confidence. These two stages are described next.

In the first stage, the procedures are analog to the coding process (Auerbach and Silverstein 2003), but with the specific goal of developing concepts and relating them according to the diagrammatic model definitions given earlier. Hence, the coding in the SSM does not necessarily need to go through a continuous and iterative process of small steps as it is usually indicated for coding, but it can be focused on the elements of the theoretical structures. There are several recommendations for performing this coding process in the SSM. For instance, one of the recommendations is the translation procedure (Britten et al. 2002). In the SSM, as the goal is to aggregate evidence by combining the compatible theoretical structures, the translation procedure can support the identification of concepts, which at first glance are not comparable, but when translated to the proper concept they become comparable. One example in software context would be translating Understandability and Learnability by a more generic concept, for instance, Usability.[1] This kind of generalization is not free from threads and should be considered in case by case basis according to the researchers' interpretations. Readers interested in a detailed view regarding the recommendations and heuristics for the coding process in the SSM can find in Santos (2015).

During the coding, besides the evidence model concepts, it is also necessary to determine the *effects* intensity and the *moderators* direction. For qualitative studies, the adverbs and adjectives used to qualify the reported outcomes are translated to the seven-point *Likert* scale describing the *effect* size or intensity. When there was no indication of the *effect* intensity we were conservative and decided to define a range of *values* to represent the imprecision regarding the intensity, e.g., 'between weakly positive and positive.' For the quantitative studies, on the other hand, we need to arbitrate ranges of *values* using the domain of the dependent variable scale as input to be able to translate it to the seven-point *Likert* scale. For instance, in Fitzgerald et al. (2014) "*the overall cycle time was reduced from almost 100 days to just over 60 days, a significant improvement*" – the authors qualified the difference of 40 days as a *significant* improvement, which was used to determine this *effect* as strongly positive in the *Likert* scale.

In the second stage, with the concepts and their relationships defined, the SSM needs further definitions to determine the confidence (i.e., the *belief value*) related to the *effects* and *moderators*. Two inputs are used to that end. One is the study type of which evidence was acquired. The SSM uses the GRADE evidence hierarchy (Atkins et al. 2004) to split the 0–1 belief value range into four subranges: unsystematic observations [0.00, 0.25]; observational studies [0.25, 0.50]; *quasi*-experiments [0.50, 0.75]; and randomized controlled [0.75, 1]. The second input is the quality assessment which is translated into the 0.25 subrange. The SSM proposes to use two checklists to assess the quality of each study, which are explained in Santos and Travassos (2013). Based on this, the *belief values* listed in Table 5 (Appendix A) are calculated, e.g., the study P1 was observational (0.25), and in the performed quality assessment using the checklists, it got 0.17 out of 0.25. As one can see, the estimation procedure give lower *belief values* for less reliable studies and higher values for the more reliable ones. Thus, the basic idea is to reflect the reliability of the evidence represented by a theoretical structure. Details regarding the quality assessment for each study can be found in the *Evidence Factory* tool.

For performing these two stages for translating evidence from the primary studies into the diagrammatic evidence models, three researchers – the first three authors – organized the tasks in the following manner.

First, the 20 papers were evenly distributed among the researchers. Each of them thoroughly read the papers and extracted the relevant information to create the evidence models. The benefits and challenges enumerated in the secondary study by Ahmad et al. (2018) were used as the primary source for identifying the *effects* (i.e., benefits) and *moderators* (i.e., challenges) in the primary studies' reports. It should be noticed that this process is usually performed inductively based on the primary studies textual report, but in the specific case of this study we used the work of Ahmad et al. (2018) as the benefits and challenges represent the codes extracted from the primary studies. Still, we were not able to find all the benefits and challenges in the papers as indicated in Ahmad et al. (2018) – the differences are presented in Section 4. On the other hand, *contextual aspects* were identified using the different SSM recommendations and heuristics.

Second, after the evidence models' creation, the researchers discussed the models together. Each researcher summarized his papers and presented the models for the other two. During this process, three mains aspects were focused: (i) assessing the understanding of the primary studies' context and outcomes, (ii) indicating the trace between the theoretical structures' concepts and the excerpts from which they were extracted, and (iii) reaching a consensus regarding the theoretical structures' concepts definition (e.g., guided by the reciprocal translation procedure indicated in the SSM – adapted from Meta-Ethnography (Da Silva et al. 2013)).

3.3 SSM step 3: Evidence synthesis

In this step, the evidence extracted from the primary studies are aggregated based on the evidence models. Therefore, it is essential to define what makes theoretical structures to match, i.e., what makes them compatible and allowing to aggregate evidence. The SSM defines two theoretical structures are compatible when their *value concepts* are the same or have the same meaning, which includes the *cause*, *archetypes*, and *contextual aspects*. Once the researcher determines that the theoretical structures can be compatible, then their *effects* and *moderators* are combined according to their directions and intensities.

Pair-by-pair comparisons determine the compatibility among theoretical structures. When a pair is found to be compatible, the combined theoretical structure is formed by the common *value concepts* of both theoretical structures being compared and by the *variable concepts* present in at least one of the two structures. *Archetypes* and *contextual aspects*, represented by *value concepts*, describe the conditions under which the aggregation is valid. For instance, in order to an evidence model be compatible with the one shown in the Fig. 1, it must have the same *value concepts*, namely: 'portfolio management,' 'software development process,' 'software project,' 'distributed project,' 'software team,' 'medium-scale system,' and 'Kanban.'

After identifying compatibility based on the *value concepts*, the *variable concepts'* (i.e., *effects* and *moderators*) intensity (e.g., positive or negative) and uncertainty (i.e., *belief value*) are pooled, in such a way that their intensity reflects the resulted agreement on the combined evidence. To that end, an uncertainty formalism is necessary to combine the results – otherwise, a simple vote counting strategy would be used. In the SSM, the Mathematical Theory of Evidence (Shafer 1976) (also known as *Dempster-Shafer* theory, DST) is the mathematical formalism that enables obtaining the pooled outcomes. The DST uses two primary inputs to combine two pieces of evidence. One is the hypotheses believed to have a chance to be true – a *belief value* greater than zero – and the other is the *belief values* themselves. Hypotheses are defined as sets of the powerset of the defined frame of discernment set whereas the *belief value* is estimated based on the procedures described in the previous step.

In order to perform the aggregation in the SSM using the DST formalisms, the different intensity values that an *effect* is possible to assume is represented as the frame of discernment. Since the intensity of an *effect* uses a seven-point *Likert* scale, the corresponding frame of discernment in the DST is defined as Θ = {SN, NE, WN, IF, WP, PO, SP} – the element names are abbreviations for the *Likert* scale terms, e.g., SN is 'strongly negative,' IF is 'indifferent,' and WP is 'weakly positive.' Likewise, the frame of discernment for *moderators* is formed by three values that are used to indicate the moderation direction: Θ = {IP, IF, DP} – 'inversely proportional,' 'indifferent,' and 'directly proportional,' respectively.

Once hypotheses and *belief values* are defined for each evidence, then the *Dempster's Rule of Combination* is applied. Eq. (1) shows that the aggregated *belief value* for each hypothesis C is equal to the sum of the product of the hypotheses' *belief values* whose intersection between all hypotheses A_i and B_j of both evidence is C. The function *m* is called *basic probability assignment function* which, as the name implies, is used to assign a *belief value* to the different hypotheses of the powerset.

$$m_3(C) = \frac{\sum_{\substack{i,j \\ A_i \cap B_j = C}} m_1(A_i)\, m_2(B_i)}{1-K}, \textit{where } K = \sum_{\substack{i,j \\ A_i \cap B_j = \varnothing}} m_1(A_i)\, m_2(B_i) \tag{1}$$

When the intersection between two hypotheses is an empty set, we say that there is a conflict. A conflict is, then, redistributed to the aggregated hypotheses – that is the function of 1 - K in the denominator. More details about how DST is used in SSM are available in Santos and Travassos (2013). At the end of Section 5, the reader finds an example of how to compute it.

4 Representation of the benefits and challenges of Kanban

Before presenting the aggregated results, in this Section we describe the coding process involved in modeling the theoretical structures and the concepts developed in that process. For the sake of readability and reasonable manuscript size, only two models are detailed since 20 theoretical structures were created. The chosen models (one is qualitative and the other quantitative) are representative of the overall set of studies. Besides, they have different levels of complexity as the quantitative investigated only three benefits and the qualitative study covers several benefits and challenges.

As stated in Section 3.2.1 (SSM Step 2), the benefits and challenges were pre-determined using those enumerated in Ahmad et al. (2018). However, since the benefits and challenges were not thoroughly defined, we possibly interpreted some of them differently. For instance, the benefits 'identify impediments to flow' and 'improve workflow' relate to each other, but the exact intended difference between them is not explicit.

Using the SSM jargon, the benefits were defined as *effects* (positive influence) and the challenges as *moderators*. Not all challenges could be interpreted as *moderators* since some of them are more associated with an intrinsic characteristic of Kanban use than an external aspect moderating the effects of its utilization. For instance, 'setting up and maintaining Kanban' is a Kanban usage aspect itself, not an external issue that whether not addressed can moderate the effects. Another example is the 'poor understanding of Kanban concepts and practices,' which is a direct Kanban use consequence, while the challenge 'lack of training' reflects an external aspect that can address the 'poor understanding of Kanban concepts and practices.' Furthermore, the SSM uses *concepts*, also denominated as *constructs*, in the *theoretical structures*. Therefore, we had to code the benefits and challenges enumerated in Ahmad et al. (2018) to *constructs.* Then, for each *construct*, we provided a definition for it based on the technical literature. For instance, 'improve quality' was coded as 'internal quality' since the quality improvement aspect reported in the primary studies was always related to an internal property of the software product. Once this interpretation was made, we provided a definition based on ISO (2017a). Also, any adjectives were removed since the effect intensity represents it – e.g., 'improve communication' became 'communication.' The output of these interpretations and considerations are shown in Table 6 (Appendix A).

Effects and *moderators* account for the *variable* concepts only. Regarding the *value concepts*, they were coded directly from the primary studies. Although most of the papers were observational and provided a relatively wealthy description of the studies' context, few of them explicitly stated what factors were determinant for the results found. Under these circumstances, the researcher constructing the *theoretical structures* needs to use their understanding of these factors to decide when they are relevant enough to be made explicit in the evidence models. An alternative to this approach would be to describe all contextual aspects reported in the studies, such as programming languages and types of products developed. However, besides adding a considerable amount of complexity to the aggregation process as all conflicts between the models need to be resolved in the *Evidence Factory* tool, the decision of whether a conflict represents inadmissible evidence (i.e., the results cannot be aggregated) would still be a researcher interpretation. Given this line of reasoning and with the goal of generalizing the results, we adopted the first approach to describe the essential contextual aspects in our interpretations. As we discuss in the next section, this keeps the aggregation in a manageable size and brings the focus to the essential contextual aspects.

Figure 1 presents evidence representation for the study P10 (Fitzgerald et al. 2014). The model has three reported benefits 'time-to-market,' 'control of project activities and tasks,' and 'continuous learning.' Also, besides the *cause* 'Kanban' and the archetypes, it has six *value concepts* describing the context: 'portfolio management,' 'software development process,' 'software project,' 'distributed project,' 'software team,' and 'medium-scale system.' The paper reports a study in a Polish company in which the researchers used a mathematical model (Erlang-C model) to gather and analyze data to improving the decision-making process regarding the Kanban process used for portfolio

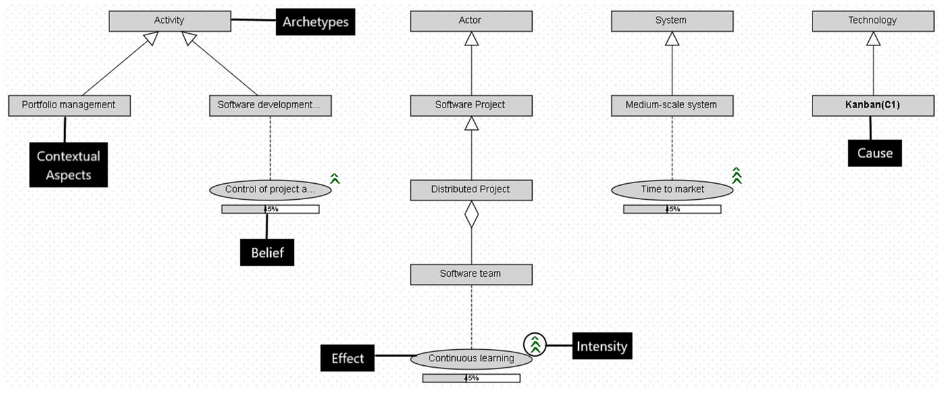

Fig. 1 Evidence model representing the results of P10

management. Projects of medium-scale systems formed the company portfolio: "*very common category of development projects at Ericpol was one where the inflow of projects was random and not controlled by the software development function, where fast response time and short service time were crucial, work effort was relatively small, and technical complexity was usually moderate*." As it can be seen, the three most salient contextual aspects are 'portfolio management,' 'distributed project,' and 'medium-scale system.' They can be used as a starting point for explanations in case of contradictory results.

From the three evidence model *effects*, two were quantitative ('time-to-market' and 'control of activities and tasks') based on the Erlang-C model and one qualitative using field observations. 'Time-to-market' was measured regarding the cycle time (in days): "*the overall cycle time was reduced from almost 100 days to just over 60 days, a significant improvement*." Moreover, 'control of activities and tasks' was extracted from the analysis of the Erlang-C model: "*the bigger an organization is (the more teams it has), the less sensitive it is to fluctuations in the average inflow … This suggests that … all the teams across different sites should work on a common input queue or backlog for software development*." Notice that for the 'time-to-market' concept the *effect intensity* was defined as {PO, SP} since it was considered a significant improvement by the authors whereas for 'control of activities and tasks' the intensity was defined as {WP, PO} since the authors indicated that it was a "suggestion" that the tasks and activities should be put on a common queue. The third, and last, *effect* 'continuous learning' was extracted from the following excerpt: "*it demonstrates how an organization can make better decisions based on data gathered and analyzed using a model (in this case, Erlang-C) that is highly relevant in the organization's context*." In this case, "highly relevant" was the qualification that to define the 'continuous learning' intensity as {PO, SP}.

The second evidence model presented in Fig. 2 is a study (Dennehy and Conboy 2017) concerned with the investigation of what the authors denominate 'flow techniques' to which Kanban, particularly the board, is directly related. According to the authors, citing (Womack and Jones 1997), flow in product development is defined as "*the progressive achievement of tasks along the value stream so that a product proceeds from design to launch, order to delivery, and raw materials into the hands of the customer with no stoppages, scrap, or backflows*." The investigation was based on the

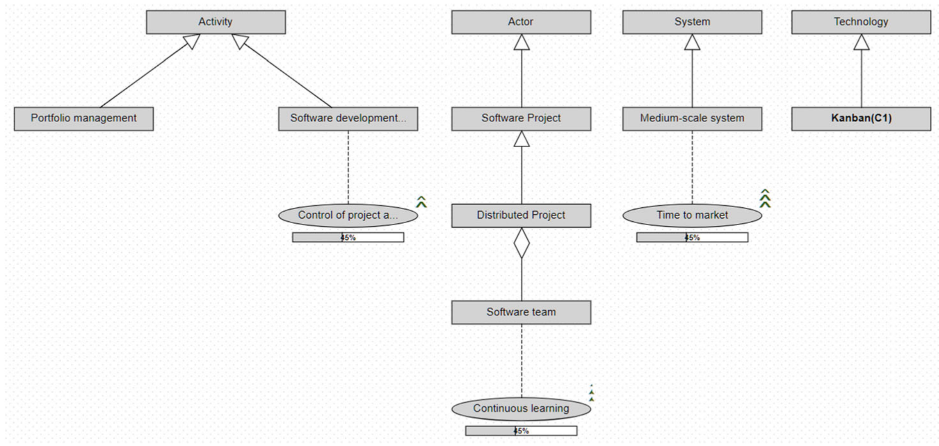

Fig. 2 Evidence model representing the results of P9

Activity Theory, which is a framework for studying different forms of human practice as historically developing cultural systems. It is used to identify the contradictions and congruencies among the six constructs of the model, namely, tools and signs, subject, object, rules and norms, community, and division of labor. This analysis, using the Activity Theory, was performed in two large software companies (with a workforce of more than 100,000 employees) in a multiple-case design with cross-case analysis.

The richness of the investigations and analysis with the Activity Theory is reflected in the evidence model. It contains five *effects*, two *moderators*, and six *contextual aspects*. For instance, the improvement of the 'flow of work,' which was the main focus of the study, was explained in the following excerpt: "*after an initial period of using the physical Kanban, a congruency in the work activity emerged between the subject and the tool, as well as the community. A manager at Company B explained that by using the physical Kanban, it was 'enacting everyone involved by basically seeing where the problems are.'*" In the coding process, 'flow of work' was always associated with the identification of impediments to the flow (see Table 6 in Appendix A). Also, as the authors indicate a complete congruency for this issue, the *effect intensity* was defined as

Table 1 Effects and moderators as reported in the selected studies

Effect / Moderator name	Representation of evidence from single studies, shown as intensity (belief value)																			
	P1	P2	P3	P4	P5	P9	P10	P11	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24
Work visibility	››› 0.42	››› 0.38	›› 0.42	››› 0.42	››› 0.40	››› **0.47**		››› 0.42	▨	››› 0.43	›› 0.44		››› 0.45		›› 0.38	››› 0.42		›› *0.22*	››› 0.46	
Control of project activities and tasks	››› 0.42	▨			››› 0.40	››› 0.47	›› *0.45*	››› 0.42	››› 0.44		›› 0.44				››› *0.19*	››› *0.21*		›› 0.45	›› *0.23*	
Flow of work	››› 0.42	▨	›› 0.42		››› 0.40	››› 0.47					›› 0.44		››› 0.45			››› 0.42		›› 0.45	▨	
Workflow		▨		››› 0.42				››› 0.42				›› 0.15			››› 0.38	››› 0.42				
Time-to-market							››› 0.45					›› 0.15							›› 0.46	›› 0.92
Task prioritization	››› 0.42		›› 0.42								›› 0.44		››› 0.45							
External Quality		▨								››› 0.43			››› 0.45				› 0.41			
Internal Quality												›› *0.07*								
Conformance										››› 0.43	▨	›› 0.15	››› 0.22							▨
Collaboration				›› 0.42		›› 0.47							▨							
Communication	›› 0.42			›› 0.42						››› **0.43**	›› **0.44**		▨							
Motivation								››› 0.42		››› **0.43**		›› *0.15*	››› 0.45		›› *0.19*	››› **0.42**				
Team Cohesion					››› 0.40								▨			‹ 0.42			▨	
Customer Satisfaction										››› 0.43	››› 0.44		››› 0.45			▨				
Continuous learning							››› 0.45					›› 0.15	››› 0.45			›› 0.42			›› *0.23*	
Strategic alignment			›› *0.21*		››› 0.40															
Expertise						↘ 0.47	▨						↘ 0.45						↘ 0.46	
Organizational Culture				↘ 0.42							↘ 0.44		↘ 0.45	↘ 0.49				↘ *0.22*		
Supporting practices										↘ 0.43	↘ 0.44	↘ 0.15								
Training				↘ 0.42		↘ 0.47				▨										▨
Managed communication between team and customer											↘ 0.44									

{PO, SP}. One last example is regarding a moderator, the 'management' 'expertise.' In P9, the possible moderation of the management expertise was described as follows: "*a deeper analysis revealed that this change could be linked to the creation of another contradiction between the tools that was caused by the introduction of flow. That is, between the expertise and knowledge of the project manager (mental tool) and the capability offered by the Kanban (physical tool). It led to a shift in rules and norms concerning the activity. Evidence of this contradiction is captured by a manager at Company B who acknowledged that he was initially skeptical of the tool (Kanban) because it 'looked very gimmicky and I just don't trust pieces of paper stuck to the wall.'*" Regarding the *value concepts*, the 'distributed project' is the only one representing a distinctive P9 study characteristic and as mentioned in the first evidence model can be used for explanations of contradictory results. Also, it is interesting to notice the absence of the 'system' *archetype* as in P9 there was no *effect*, *moderator* or *contextual aspect* related to it.

These two evidence models give a glimpse of how the coding process regarding the Kanban studies was performed using the SSM. For completeness, we show in Table 1 all the *effects intensities* and *moderators directions*, in addition to their *belief values*. Given our understanding and interpretation of the benefits and challenges (Table 6 in Appendix A), their identification in the primary studies diverged from the ones indicated in Ahmad et al. (2018). Shaded cells indicate that these were indicated as a benefit or challenge for the respective study in Ahmad et al. (2018), but we could not identify it in our study. *Belief values* in italic font represent the benefits or challenges that we were not confident about their identification in the respective study – for those cases; we applied a 50% discount on the *belief value*. Moreover, the ones in bold font are those which we identified in the respective study, but they were not indicated in Ahmad et al. (2018). Apart from that, the *effects intensities* were represented using the notation presented in Section 3.2; and the moderators use ↗ for directly proportional and ↘ for inversely proportional.

5 Results

The aggregation was performed following the procedures described in Section 3.3. We first present the results of the primary aggregation focus, which is the pooled *effects* and *moderators*. Then, at the end of this Section, we detail some aspects of the aggregation process such as how the pooled results were computed and how the evidence models compatibility was analyzed.

Table 2 shows the results after performing the aggregation of evidence on the benefits and challenges of using Kanban in SE. Apart from this section as a whole, Table 2 synthesizes the answer to the research question defined at the beginning of Section 3. The first column shows the reported *effect* (i.e., benefit) caused by or the *moderator* (i.e., challenge) influencing the introduction of Kanban in the organization. The second column indicates the primary studies that have reported this *effect* and the third the number of papers. The fourth column shows the aggregated *effect intensity* about how the use of Kanban causes such *effect* (e.g., positive or negative). The fifth column represents the aggregated *belief* of such *effect*. It is one of the most exciting results of the aggregation. Table 2 shows in bold the effects and moderators in the upper quartile (Q3) and underlined those equal or higher than the median (Q2) after the aggregation – the quartile calculation was separated for *effects* (Q2 = 75.5; Q3 = 93.75) and *moderators* (Q2 = 73; Q3 = 85). The sixth column shows whether there was a conflict while

Table 2 Aggregated effects and moderators of Kanban use

Effect/Moderator related to Kanban use	**Representation of evidence from single studies, shown as intensity (belief value)**					
	Paper ID	***Number of papers***	***Intensity***	***Belief***	***Conflict***	***Difference***
Work Visibility	P1, P2, P3, P4, P5, P9, P11, P14, P15, P17, P19, P20, P22, P23	14	>>>	**99%**	-	52%
Control of project activities and tasks	P1, P5, P9, P10, P11, P13, P15, P19, P20, P22, P23	11	>>>	**96%**	-	49%
Flow of work	P1, P3, P5, P9, P15, P17, P20, P22	8	>>>	**94%**	-	47%
Workflow	P4, P11 P16, P19, P20	5	>>>	93%	-	51%
Time-to-market	P10, P16, P21, P23, P24	5	>>	**96%**	-	4%
Task prioritization	P1, P3, P15, P17	4	>>>	68%	-	23%
External quality	P14, P17, P21	3	>>>	68%	-	23%
Internal Quality	P16	1	>>	7%	-	0%
Conformance	P14, P16, P17	3	>>>	55%	-	12%
Communication	P1, P4, P14, P15	4	>>	81%	-	37%
Collaboration	P4, P9	2	>>	70%	-	23%
Motivation	P11, P14, P16, P17, P19, P20	5	>>>	89%	-	44%
Team cohesion	P5, P20	2	<	31%	0.16	-11%
Customer satisfaction	P14, P15, P17	3	>>>	82%	-	37%
Continuous learning	P10, P16, P17, P20, P23	5	>>>	69%	-	23%
Strategic alignment	P3, P5	2	>>	52%	-	12%
Expertise	P9, P17, P23	3	↝	77%	-	30%
Organizational culture	P4, P15, P17, P18, P22	5	↝	**93%**	-	44%
Supporting practices	P14, P15, P16	3	↝	73%	-	29%
Training	P4, P9	2	↝	69%	-	22%
Managed communication between team and customer	P15	1	↝	44%	-	0%

aggregating that *effect*. It is essential to analyze and to characterize different contexts from which the evidence was gathered. Lastly, the seventh column shows the difference between the *belief* max value in individual papers and the gained confidence after the aggregation. Therefore, a positive difference indicates the effects that have been reinforced after the aggregation whereas a negative difference shows that the evidence is somewhat contradictory.

The aggregation indicates that 'work visibility,' 'control of project activities and tasks,' 'flow of work,' and 'time-to-market' are the main benefits of Kanban by using the criteria of *belief values* in the upper quartile. The evidence regarding these benefits is vast and consensual. Among the *moderators*, a Kanban successful adoption is most conditioned to the 'organizational culture.' It should be noticed that we ignored the *effect intensity* selection strategy used by default in the SSM to use the quartile criteria. The strategy tries to balance outcomes precision and confidence, by selecting a singleton (a value in the *Likert* scale) or a compound (a range in the *Likert* scale) hypothesis as an aggregation result. The use of this strategy is just a suggestion of the SSM since, as discussed in Santos and Travassos (2013), this is a definition related to the Mathematical Theory of Evidence and there is no consensual way to perform this selection (Bloch 1996). With this default strategy, we would have three different results in Table 2: (i) 'work visibility' with PO intensity and 84% of *belief*, (ii) 'control of project activities and tasks' with PO intensity and 83% of *belief*, and (iii) 'flow of work' with PO intensity and 77% of *belief*. These results are more precise, but with the tradeoff of lower confidence. As in the case of this synthesis, all *effects* in the evidence models were modeled using an intensity range (see Section 3.2.1), then we opted to keep the range for the outcomes as well despite the default strategy of the SSM. Still, it shows that for these three *effects*, contrarily to the others, there is a significant amount of *belief* assigned to the singleton. It demonstrates how the Mathematical Theory of Evidence converges with the accumulation of evidence. For instance, 'flow of work' has three evidence models (P3, P15, and P22) with {WP, PO} intensity and five models (P1, P5, P9, P17, and P20) with {PO, SP} intensity. As one can see, {PO} is the intersection between the two ranges.

Kanban {positively – strongly positively} affects the 'work visibility' of software development projects. Indeed, Kanban is by definition regarded as a visualization tool to introduce Lean ideas. "*The Kanban board offers better transparency of the development process and shows which developer is working on which task*" (P20). Comparing to Scrum, "*the difference is the Scrum board resets between each iteration while the Kanban board is normally persistent and doesn't need to reset and start over. Further, tasks are visualized on the Scrum board for each sprint, while Kanban visualizes tasks that can be pulled at any time to respect WIP limits; this restricts the allowed number of tasks in every workflow state*" (P1). Moreover, even in the educational environment, the "*overall perception of students (55%) about the Kanban board was positive. It helps in visualizing and prioritizing an entire work project more efficiently*" (P2).

Kanban {positively – strongly positively} affects the 'flow of work' of the software development process. Associated with the 'work visibility,' the 'flow of work' is another Kanban core aspect as it "*provides greater visibility into what teams are doing... improving the feedback loops [and] exposing resource constraints and even capacity utilization*" (P9). Thus, it can support organizations in making its value stream as efficient as possible avoiding any impediments or overworking. As a result, team members become aware that "*organizational entities cannot work independently because the outcome is related to their cooperative capabilities to create value. Interacting components are important for reaching smooth value streams and avoiding local optimizations*" (P22).

The use of Kanban {positively – strongly positively} affects the 'control of project activities and tasks' of software development projects. One crucial aspect of this improved control is focusing the work conducted by the software team since the "*work in progress limit helps teams to avoid working on too many parallel tasks and are forced to work on those tasks that deliver value to the project*" (P1). This focus is important even in higher levels of management for managing the work of more than one team as stated in P10: "*all the teams across different sites should work on a common input queue or backlog for software development.*"

The use of Kanban {weakly positively – positively affects} the 'time-to-market' of software products. It was one of the few *effects* having quantitative data since it has a natural surrogate which is a cycle or lead time. In P16, the "*the great majority of teams reduced their average lead time, some of them for more than 30%*". Kanban helps the features to be released as soon as possible as in the case of P23 in which "*variation in delivery times reduced by 78% from 30.5 to 6.8, and the mean time to develop fewer and smaller software features declined by 73% from 9.2 to 2.5 working days.*"

Other benefits that can be expected from Kanban – *belief value* equal to or higher than the median – are 'workflow,' 'communication,' 'motivation,' and 'customer satisfaction.' These represent *effects* that are likely to be observed on adopting Kanban in software organizations. Moreover, an appropriate level of 'expertise' and 'supporting practices' – usually agile practices such as test-driven development and pair programming – need close attention to operate Kanban in its full extent. The other *effects* and *moderators* have a relatively lower *belief value* and seem to not appear in practice as fewer studies reported them, or they are not completely congruent. Not complete congruency means that results are not the same, but they have an intersection – for instance, the intensities for 'external quality' {WP, PO} in P14, {WP, PO} in P17, and {IF, WP} in P21.

It is also noticeable the virtual absence of conflicts in the aggregation. A conflict occurs when the intersection between two results is empty, such as between {WN, IF} and {PO, SP} as occurred for 'team cohesion' in the evidence models of papers P20 and P5, respectively. The reasons for this are twofold. First, it is related to how we opted to model the *effect intensity*. As almost all primary studies are observational and qualitative, the benefits descriptions were not precise enough to define a single value in the seven-point *Likert* scale. Thus, the aggregation of range intensities is less susceptible to conflict. Second, it is due to research nature on this topic. The point in question is it seems that negative results are not being published, which represents an important limitation of the current body of knowledge regarding the use of Kanban in SE. The only conflict presented in Table 2 was associated precisely with a negative influence of Kanban to the 'team cohesion' in the study P20.

As described in Section 3.3, the aggregation was performed using the Dempster's Rule of Combination (see eq. 1). To illustrate how it is computed, let's take as an example the 'team cohesion' *effect*. Table 3 below is a schematic representation for computing the Dempster's Rule of Combination using eq. 1. Notice how the conflict is redistributed among the hypotheses. The highest *belief value* (excluding the frame of discernment {Θ} itself) is assigned to {WN, IF} with 0.306.

The values for the combined $m_{\text{P5-team cohesion}} \oplus m_{\text{P20-team cohesion}}$ are:

$\kappa = 0.168$ and $1 - \kappa = 0.832$,

$m_{P5} \oplus m_{P20}$ ({PO,SP}) = 0.229/0.832 = 0.275,

$m_{P5} \oplus m_{P20}$ ({WN,IF}) = 0.255/0.832 = 0.306,

$m_{P5} \oplus m_{P20}$ ({Θ}) = 0.348/0.832 = 0.418,

$m_{P5} \oplus m_{P20}$ is 0 for all other sets of the powerset of Θ.

Another vital aggregation process aspect is the determination of evidence compatibility. All the studies outcomes were considered compatible and for this reason, aggregated. To reach this conclusion the general orientation was to seek generalization. It was only possible due to the relatively high number of studies. Thus, we first assumed generalization, and if the results were conflicting, then explanations would be sought. Table 4 enumerates the generalizations applied in the aggregation. All the different kinds and size of software systems, such as 'web system' and 'large-scale system,' were ignored. Another important generalization was related to the primary goal of using Kanban. Most papers report the utilization of Kanban in a typical software development environment, i.e., in the commercial or controlled construction of software products. However, five studies were very distinct on the Kanban use primary goal. Two of them investigated the use of Kanban in

Table 3 Combination of two basic probability assignment functions (for 'team cohesion')

mP5-team cohesion \ mP20-team cohesion	{WN,IF} (0.423)	Θ (0.577)
{PO,SP} (0.397)	Ø (0.168)	{PO,SP} (0.229)
Θ (0.603)	{WN,IF} (0.255)	Θ (0.348)

Table 4 Applied generalizations in the aggregation process

Paper ID	Contextual aspect removed from the aggregated evidence model	Contextual aspect merged with another concept in the aggregated evidence model
P1	'Maintenance'	-
P2	'Education'	-
P3	'Portfolio management.'	-
P4	-	-
P5	'Education'	-
P9	'Distributed project.'	-
P10	'Portfolio management.' 'Medium-scale system.' 'Distributed project.'	-
P11	-	-
P13	'Web System'	-
P14	-	'CS & SE students' ➔ 'Software team.'
P15	-	-
P16	'Education' 'Small-sized system.'	-
P17	'Large-scale web system.'	-
P18	-	-
P19	'Distributed project'	'Scrumban' ➔ 'Kanban'
P20	-	-
P21	'Large-scale system.'	-
P22	'Distributed project.'	-
P23	'Large-scale web system.'	-
P24	'Medium-scale web system.'	-

'portfolio management,' and three used Kanban in 'education' as a tool for learning software concepts. Some studies were also conducted in a 'distributed project' setting and one with the specific purpose of software 'maintenance.'

6 Discussion

Despite the important differences in the studies' context, the aggregation did not present significant conflict in the results. The resulting aggregated evidence model indicates that Kanban main benefits are 'work visibility,' 'control of project activities and tasks,' 'flow of work,' and 'time-to-market' regardless the type of software systems under development or whether they are collocated or distributed. Also, the same results are achieved in the specific settings of 'portfolio management' and 'education.' Still, the software organizations need to have close attention to its 'organizational culture,' the 'expertise' of their management, and the right set of 'supporting practices' in place to obtain these improvements.

This work contributes to the body of knowledge on using Kanban in SE by strengthening evidence of its benefits and challenges. For most of the benefits (*effects*) and challenges (*moderators*), the results followed the trends indicated in the previous investigations. This observation helped to see what *effects* are general to different settings in which Kanban is used in software organizations. We believe that the aggregated results point to more generalized perceptions and stronger indications of its applicability. Thus, it is expected that practitioners benefit from these indications to support their decision making regarding Kanban in practice. Besides decision making, the synthesis indications provide the basis to detect when Kanban is not producing the expected outcomes, allowing software organizations to act upon accordingly. Also, it may indeed be the case that the benefits could not be observed in some settings. These situations should be documented appropriately and incorporated into the synthesis to keep it up to date.

Previous primary studies already reported the *effects* caused by and *moderators* influencing the use of Kanban. However, most of them bring detailed qualitative observations from interviews and surveys while few others present quantitative data regarding specific aspects of 'time-to-market' and 'external quality.' Hence, this synthesis, besides strengthening evidence of Kanban benefits and challenges, also presents a concise and organized view of previous works regarding this theme.

On the other hand, previous secondary studies focused on different aspects of the body of knowledge than this synthesis study. Corona and Pani (2013) cover how Kanban boards are used, what is typically represented, and which tools are being employed to support software development teams. They identified and analyzed the most commonly represented activities in Kanban boards. Nine activities were identified from all the boards analyzed, namely Specification, Analyze, Build, Development, Test, Acceptance, Deploy/Delivery, Release Ready, and Documentation. Al-Baik and Miller (2015) is mostly a conceptual paper focused on describing how the main elements are discussed in the technical literature and how they relate to the lean thinking. Based on 37 studies the authors identified 20 different concepts, principles, and techniques related to Kanban, such as 'pull system,' 'prioritized queue,' 'done item,' and 'validated learning.' As mentioned in Section 3.1, Ahmad et al. (2013) was updated in Ahmad et al. (2018). And the last paper (Ahmad et al. 2018) is a mapping study showing, as a typical study of this kind, a broad overview of Kanban publications, including their venues, number of published papers over the years, and the research methods used in the studies.

Besides the aspects previously mentioned, both Al-Baik and Miller (2015) and Ahmad et al. (2018) dedicate part of their reports to enumerate and analyze the Kanban reported benefits and challenges, which is directly related to this synthesis study. Not by chance, these two works were used as the basis for the identification of the primary studies and as the source for the aggregated benefits and challenges. However, since the benefits and challenges were not the sole goals of these two works, the way that these issues were addressed is relatively more straightforward compared to this synthesis. They used an approach that can be directly related to the vote-counting strategy, which limitations are broadly discussed in the technical literate (Pickard et al. 1998). Thus, this synthesis study represents an additional contribution to these secondary studies.

Turning the attention back to the aggregated results, although the synthesis shows the main benefits of Kanban, it should be noticed that there are several other factors with fewer studies investigating them. Thus, either they rarely appear in practice (and for this reason not considered in the investigations), or they still need further studies to improve confidence in them. Particularly regarding the effect 'team cohesion' we should add our interpretations for this divergence. The studies P5 and P20 have very distinct context. P5 was conducted in an educational environment whereas P20 in a large software process improvement initiative in a software organization. Interestingly enough, the negative intensity for 'team cohesion' came from the second study. The authors of P20 did not present their thoughts about this specific result since this was part of a questionnaire answered by the organization employees. The question was put in a 'social factors' section of a questionnaire. The only consideration in P20 was that "*process transition has impacted the social factors the least.*" Participants of P5, on the other hand, were more open for collaboration and obtaining new skills as this was precisely the capstone course goal in which the study was conducted. Using Kanban resulted in "*building positive relationships among team members, effective task management, development of a shared vision and responsibility sharing.*" We hypothesize that Kanban can help to improve social factors such as 'team cohesion,' but for this to happen the team members need to be open for the new mindset imposed by the Lean thinking and Kanban concepts.

7 Threats to validity

Aggregating qualitative and quantitative evidence is by nature an endeavor subject to different risks. To mitigate this significant threat a method appropriate for that goal the SSM has been used. The SSM method aims to support the SE community to construct and consolidate empirically-grounded knowledge (Santos and Travassos 2013). It has been used in different research topics such as software reference architecture (Martinez-Fernandez et al. 2015), software productivity (Chapetta 2016), and software inspections (Santos and Travassos 2017b). This section discusses possible threats to validity and emphasizes the mitigation actions used.

For the threat of missing critical primary studies, we used two secondary studies as a source of primary studies. We obtained a set of 20 studies reporting evidence on Kanban, which is a high number in SE considering that they report the same effects (i.e., benefits and drawbacks of Kanban). During this process, we discarded studies that only reported opinions, rather than empirically-grounded evidence.

We are aware that each selected study poses its validity threats; therefore, we carefully assessed them together with the studies' context to interpret their results appropriately. Furthermore, while representing empirical evidence from individual studies, researchers can reflect their own opinion and, thus, bias the representation. A researcher first prepared the definition and analysis of each evidence model from each selected primary study and validated together with two other researchers to mitigate these subjective issues. During this process we experienced some semantic issues, meaning that different studies referred to the same concept using different terms. This would lead to a wrong aggregation. To avoid this, we created a glossary of terms that was represented in the evidence models and kept track of the matching terms.

To improve the aggregated evidence interpretation, we used some suggested strategies (Santos and Travassos 2013). For instance, we recorded the diverse context of each study, so we could better reflect and understand the aggregated evidence. It is important to note that our aggregated results are based on what the authors reported in their papers. Hence, there is always the risk that valuable information might not have been reported. Also, even although Al-Baik and Miller (2015) and Ahmad et al. (2018) indicate what papers report each benefit and challenge, we read each paper to seek them again. It generated some divergences as listed in Table 1 to which we attribute the definitions of the benefits and challenges as listed in Table 6 (Appendix A).

Our results show that some effects got higher degrees of *belief* while others did not. Being aware of the correct interpretation of these results is essential. On the one hand, the *effects* and *moderators* that got higher *belief* are potentially those that have been further studied and agreed among the studies. On the other hand, those *effects* that got lower *belief values* (or even negative) are those that were just partially approached by the available evidence (or got contradictory results among the studies). Therefore, these *effects* are relevant topics that need to be further studied. We highly encourage the SE community interested on using Kanban to investigate the *effects* that do not have a high confidence value yet, to increase knowledge and consolidate the beliefs of the benefits and challenges of using Kanban in SE.

8 Conclusions

This paper presented a synthesis study regarding the benefits and challenges of using Kanban in SE. Despite being a relatively recent research topic, there is an extensive amount of available evidence regarding this topic, which supported the achievement of high confidence for several benefits and challenges worked out in the synthesis.

The primary contribution of this paper is to present a condensed view of the benefits and challenges regarding Kanban use in software organizations as reported in primary studies in the field. It brings an objective indication of what are the more relevant ones considering the knowledge available in the technical literature, which is an essential result of a synthesis study in comparison to the individual results of the primary studies that were aggregated. Also, it strengthens the understanding of which aspects Kanban can improve in software organizations and what factors should be addressed to achieve the best results considering all studies as a whole.

The benefits ‘work visibility,’ ‘control of project activities and tasks,’ ‘flow of work,’ and ‘time-to-market’ indeed appear to be the ones intrinsically linked to the Lean thinking and the Kanban approach. Moreover, given the synthesis results, they do seem to be present in software projects as well. Still, the results must be taken with caution. We missed primary studies with negative results regarding Kanban implementations in software organizations. According to Staats et al. (2011), failed implementations are common outside the manufacturing realm, and it is quite surprising that the secondary studies used to select the primary studies for this synthesis did not find any.

The synthesis reported in this paper can be evolved from the point where its scope has been delimited. All the data is available in the *Evidence Factory* tool, and the SSM allows to add new primary studies as necessary. One exciting addition would be the inclusion of the 23 experience reports listed in Ahmad et al. (2018). It can reinforce some of the effects and moderators which lacks evidence and can bring new insights regarding the Kanban use in software organizations.

9 Endnotes

[1]Understandability, Learnability and Usability terms were taken from the software quality terminology presented in Kitchenham and Pfleeger (1996). The possibility of generalizing Understandability and Learnability by Usability was also taken from there.

10 Appendix A

Table 5 Primary studies

Study Id	Study Type: Instruments	Participants	Company/ Environment Characteristics	Application Domain	Belief[b] & Evidence Type	Development/ Portfolio/ Maintenance	Year
P1	Observational Multiple Case Study Interviews (Skype and Face-to-face)	2 Product Owners 3 Coaches 3 Project Managers 3 Scrum Masters 6 Developers	Two Large Multinational Finnish Software Companies	Businesses for the New Digital Economy	42% Qualitative	Maintenance	2016
P2	Observational Online Survey	19 Master Degree Students	University of Oulu Software Factory	Not Specified	38% Quantitative and Qualitative	Education	2014
P3	Observational Interview (Audio and Transcript)	1 Director Of R&D 1 Senior Manager 1Head of Quality and Environment 1 CEO 1 Business Manager 1 Senior Consultant 1 Internal Coach 1 Agile Coach	Agile and Lean Finnish Software Companies	Computer Security, Embedded Systems, IT Services, Telecon Network, Telecon, Consultancy	42% Qualitative	Portfolio	22,017
P4	Observational Online Survey	146 from LinkedIn LeanKanban Incorporated group 27 Organizations	LinkedIn Group	IT Services Hardware Manufacturing Telecommunication	42% Quantitative and Qualitative	Development	2016
P5	Observational Online Survey	51 Master Degree Students	University of Oulu Software Factory	Not Specified	40% Quantitative and Qualitative	Education	22,014
P9	Observational Multiple Case Study Interview	15 Various Roles IT Director Portfolio Manager System Integration Manager Financial Controller	Two Companies Multinational	Technology Services Telecommunication	47% Qualitative	Development	22,017
P10	Observational Case Study	467 Software Projects Over 22 Month Period	Ericpol Engineering Company Multinational	Communications Healthcare Finance	45% Quantitative	Portfolio	22,014
P11	Observational Action Research Questionnaire Weekly Notes Personal Notes	8 Students	Not Specified	Open Software Project	43% Qualitative	Maintenance	22,016
P13	Observational Semi-Structured Interview	13 Master Degree Students	University Of Helsinki R&D Software Factory	Web Services	44% Qualitative	Development	22,010
P14	Observational Case Study	12 Master Degree Students	University Of Helsinki R&D Software Factory Educational	Business Prototype Web Application	43% Qualitative	Development	22,011

Table 5 Primary studies *(Continued)*

Study Id	Study Type: Instruments	Participants	Company/ Environment Characteristics	Application Domain	Belief[b] & Evidence Type	Development/ Portfolio/ Maintenance	Year
P15	Observational Semi-Structured Interview	10 Software Development Practitioners	Iceland Industry	Agile Consultancy Telecommunication Web Services	44% Qualitative	Development	22,015
P16	Non-Systematic Study	62 Undergraduate Students 16 Teams	University of Ljubljana	Educational Undergrad Students	15% Quantitative and Qualitative	Education	2015
P17	Observational Exploratory Case Study	9 Staff Project Manager Business Analyst Software Architect Tester Lead Developer	BBC Worldwide London	Web Development	44% Quantitative	Development	22,012
P18	Observational Case Study Semi-Structured Interviews	7 Software Practitioners 2 Software Developer 1 Scrum Master 1 Area Product Owner 1 Kanban Consultant 1 Software Specialist 1 Senior Manager	Two Large Scale Companies from Northern Europe	Not Specified	49% Qualitative	Development	22,015
P19	Observational Action Research Unstructured Interviews Email Online Conversations	15–20 Developers	Vietnamese Office of Swedish Software Company CMSiPro	Content Management System	38% Qualitative	Development	2012
P20	Observational Action Research Interviews Email Online Conversations	15–20 Developers	Swedish Office of Swedish Software Company CMSiPro	Content Management System	42% Qualitative	Development	22,011
P21	Observational Case Study	100 Developers and Specialists	Multinational Scandinavian Software Company	Not Specified	41% Quantitative	Development	22,012
P22	Observational Case Study Focus Group Survey	49 Employees	Elektrobit Finnish Software Company	Wireless and Embedded Systems	44% Qualitative	Development	22,014
P23	Observational Synthesis	9 Staff Project Manager Business Analyst Software Architect Tester Lead Developer	BBC Worldwide London	Web Development	46% Quantitative	Development	22,011
P24	Randomized Controlled Simulation	10 Developers	Simulation	Not Specified	92% Quantitative	Development	22,011

Table 6 Effects and moderators' definitions

Benefit – as in Ahmad et al. (2018)	Effect	Description
Improve visibility and transparency	Work visibility	Work is visible when any software project member is able to obtain the information necessary to determine the state of the project tasks.
Better control of project activities and tasks	Control of project activities and tasks	The availability or implementation of tools, techniques, or practices to handle, avoid, or mitigate unpredictable tasks during the software development.
Identify impediments to flow	Flow of work	The flow of work through the software development process.
Improve workflow	Workflow	An orchestrated and repeatable pattern of business activity enabled by the systematic organization of resources into processes that transform materials, provide services, or process information. (ISO 2017b)
Faster time-to-market	Time-to-market	Time-to-market is the strategy of focusing on reducing the time to introduce new products to market. (Pawar et al. 1994)
Improve prioritization of products and tasks	Task prioritization	As a principle, it means doing 'first things first;' as a process, it means evaluating a group of items and ranking them in their order of importance or urgency. (Barney 1986)
Decrease defects and bugs	External Quality	A measure of the degree to which a software product enables the behavior of a system to satisfy stated and implied needs when the system including the software is used under specified conditions. NOTE: Attributes of the behavior can be verified and/or validated by executing the software product during testing and operation.(ISO 2017a)
Improve quality	Internal Quality	A measure of the degree to which a set of static attributes of a software product satisfy stated and implied needs when the software product is used under specified conditions. NOTE 1: Static attributes include those that relate to the software architecture, structure, and its components. NOTE 2: Static attributes can be verified by the review, inspection, simulation and/or automated tools. (ISO 2017a)
A lightweight, intuitive method	Conformance	The concept of an agreement between a process and its model is what is referred to as process conformance.
Improve communication and collaboration	Collaboration	To work jointly with others or together especially in an intellectual endeavor. (Stevenson 2010)
Improve communication and collaboration	Communication	The imparting or exchanging of information by speaking, writing or using some other medium.
Improve team motivation	Motivation	The personnel or team motivation may come from job satisfaction, job involvement, and organizational commitment.
Team building and cohesion	Team Cohesion	According to COCOMO II, the Team Cohesion accounts for the sources of project turbulence and entropy due to difficulties in synchronizing the project's stakeholders: users, customers, developers, maintainers, interface designers, others. These difficulties may arise from differences in stakeholder objectives and cultures; difficulties in reconciling objectives; and stakeholder's lack of experience and familiarity in operating as a team.
Increase customer satisfaction	Customer Satisfaction	Customer satisfaction is a perception. It's also a question of degree. It can vary from high satisfaction to low satisfaction. If customers believe that you've met their requirements, they experience high

Table 6 Effects and moderators' definitions *(Continued)*

Benefit – as in Ahmad et al. (2018)	Effect	Description
		satisfaction. If they believe that you've not met their requirements, they experience low satisfaction. (ISO 2005)
Promoting a culture of continuous learning	Continuous learning	The process by which individual and/or organizational learning is fostered on an ongoing basis. (Tannenbaum 1997)
Strategic alignment	Strategic alignment	Alignment is related to the establishment of a shared view about software features between stakeholders. Alignment is usually necessary at any level of software development. For instance, developers should adopt the same perspective when refactoring source code. Alignment is also necessary between information systems departments and the business as a whole.
Challenges – as in Ahmad et al. (2018)	Moderator	Description
Setting up and maintaining Kanban	This challenge does not represent a moderator.	
Management not ready for new method	Expertise	Expert skill or knowledge in a particular field. (Stevenson 2010)
Poor understanding of Kanban concepts and practices	This challenge does not represent a moderator.	
Managed communication between teams and customer	Managed communication between teams and customer	The imparting or exchanging of information by speaking, writing or using some other medium. (Stevenson 2010)
Changing organizational culture	Organizational culture	A complex set of values, beliefs, assumptions, and symbols that define the way in which a firm conducts its business. (Barney 1986)
Lack of supporting practices around the use of Kanban	Supporting practices	Supporting practices necessary or essential for the use of a method, technique or procedure.
Lack of training	Training	The process of learning the skills you need to do a particular job or activity. (Stevenson 2010)
Poor knowledge management	This challenge was present only in one study, which was not included in the synthesis.	

10.1 Appendix B. Primary studies

The studies P5, P6, P7, and P12 were not included in the synthesis because of the reasons explained in Section 3.1. However, they were kept on this list for the IDs continuity since they were referenced in the Evidence Factory tool.

P1. Ahmad MO, Kuvaja P, Oivo M, Markkula J (2016) Transition of Software Maintenance Teams from Scrum to Kanban. In: 2016 49th Hawaii International Conference on System Sciences (HICSS). pp. 5427–5436.
P2. Ahmad MO, Liukkunen K, Markkula J (2014) Student perceptions and attitudes towards the software factory as a learning environment. In: 2014 IEEE Global Engineering Education Conference (EDUCON). pp. 422–428.
P3. Ahmad MO, Lwakatare LE, Kuvaja P, Oivo M, Markkula J (2016) An empirical study of portfolio management and Kanban in agile and lean software companies. Journal of Software: Evolution and Process.

P4. Ahmad MO, Markkula J, Oivo M (2016) Insights into the Perceived Benefits of Kanban in Software Companies: Practitioners' Views. In: Agile Processes, in Software Engineering, and Extreme Programming. Springer, Cham, pp. 156–168.

P5. Ahmad MO, Markkula J, Oivo M (2014) Kanban for software engineering teaching in a software factory learning environment. World Transactions on Engineering and Technology Education 12:338–343.

P6. Ahmad MO, Markkula J, Oivo M (2013) Kanban in software development: A systematic literature review. In: 2013 39th Euromicro Conference on Software Engineering and Advanced Applications. pp. 9–16.

P7. Al-Baik O, Miller J (2015) The Kanban Approach, Between Agility and Leanness: A Systematic Review. Empirical Softw Engg 20:1861–1897. doi: https://doi.org/10.1007/s10664-014-9340-x

P8. Corona E, Pani FE (2013) A review of lean-kanban approaches in the software development. Transactions on Information Science and Applications: 10(1):1–13.

P9. Dennehy D, Conboy K (2017) Going with the flow: An activity theory analysis of flow techniques in software development. Journal of Systems and Software 133:160–173. doi: https://doi.org/10.1016/j.jss.2016.10.003

P10. Fitzgerald B, Musiał M, Stol K-J (2014) Evidence-based Decision Making in Lean Software Project Management. In: Companion Proceedings of the 36th International Conference on Software Engineering. ACM, New York, NY, USA, pp. 93–102.

P11. Harzl A (2016) Combining FOSS and Kanban: An Action Research. In: Open Source Systems: Integrating Communities. Springer, Cham, pp. 71–84.

P12. Heikkilä VT, Paasivaara M, Lassenius C (2016) Teaching University Students Kanban with a Collaborative Board Game. In: 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C). pp. 471–480.

P13. Ikonen M, Kettunen P, Oza N, Abrahamsson P (2010) Exploring the Sources of Waste in Kanban Software Development Projects. In: 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications. pp. 376–381.

P14. Ikonen M, Pirinen E, Fagerholm F, et al. (2011) On the Impact of Kanban on Software Project Work: An Empirical Case Study Investigation. In: 2011 16th IEEE International Conference on Engineering of Complex Computer Systems. pp. 305–314.

P15. Law EL-C, Lárusdóttir MK (2015) Whose Experience Do We Care About? Analysis of the Fitness of Scrum and Kanban to User Experience. International Journal of Human-Computer Interaction 31:584–602. doi: https://doi.org/10.1080/10447318.2015.1065693

P16. Mahnic V (2015) From Scrum to Kanban: Introducing Lean Principles to a Software Engineering Capstone Course. International Journal of Engineering Education 31:1106–1116.

P17. Middleton P, Joyce D (2012) Lean Software Management: BBC Worldwide Case Study. IEEE Transactions on Engineering Management 59:20–32. doi: https://doi.org/10.1109/TEM.2010.2081675

P18. Tripathi N, Rodríguez P, Ahmad MO, Oivo M (2015) Scaling Kanban for Software Development in a Multisite Organization: Challenges and Potential Solutions.

In: Agile Processes in Software Engineering and Extreme Programming. Springer, Cham, pp. 178–190.

P19. Nikitina N, Kajko-Mattsson M, Strâle M (2012) From scrum to scrumban: A case study of a process transition. In: 2012 International Conference on Software and System Process (ICSSP). pp. 140–149.

P20. Nikitina N, Kajko-Mattsson M (2011) Developer-driven Big-bang Process Transition from Scrum to Kanban. In: Proceedings of the 2011 International Conference on Software and Systems Process. ACM, New York, NY, USA, pp. 159–168.

P21. Sjøberg DIK, Johnsen A, Solberg J (2012) Quantifying the Effect of Using Kanban versus Scrum: A Case Study. IEEE Software 29:47–53. doi: https://doi.org/10.1109/MS.2012.110

P22. Rodríguez P, Partanen J, Kuvaja P, Oivo M (2014) Combining Lean Thinking and Agile Methods for Software Development: A Case Study of a Finnish Provider of Wireless Embedded Systems Detailed. In: 2014 47th Hawaii International Conference on System Sciences. pp. 4770–4779.

P23. Senapathi M, Middleton P, Evans G (2011) Factors Affecting Effectiveness of Agile Usage – Insights from the BBC Worldwide Case Study. In: Agile Processes in Software Engineering and Extreme Programming. Springer, Berlin, Heidelberg, pp. 132–145.

P24. Cocco L, Mannaro K, Concas G, Marchesi M (2011) Simulating Kanban and Scrum vs. Waterfall with System Dynamics. In: Agile Processes in Software Engineering and Extreme Programming. Springer, Berlin, Heidelberg, pp. 117–131.

Abbreviations

DP: Directly proportional; DST: Dempster-shafer theory; IF: Indifferent; IP: Inversely proportional; NE : Negative; PO: Positive; SE: Software engineering; SN : Strongly negative; SP: Strongly positive; SSM: Structured synthesis method; WN: Weakly negative; WP: Weakly positive

Acknowledgments

Not applicable.

About the authors

Paulo Sérgio M. dos Santos holds a D.Sc. in Systems Engineering and Computer Science from COPPE/UFRJ. His research interests are primarily concerned with scientific knowledge representation and its relation to research synthesis. Also, he investigates continuous experimentation in the particular context of the internet of things. He participates in several program committees, such as the Workshop on Managing Quality in Agile and Rapid Software Development Processes, the International Conference of the Chilean Computer Science Society, and the Brazilian Symposium on Software Quality.
Alessandro C. Beltrão is currently cursing a Master's degree on the topic of Experimental Software Engineering at COPPE/UFRJ. He's graduated in Software Engineering at the University of Brasilia and had experience as an exchange student at Auckland University of Technology.
Bruno P. de Souza is a Master's student in the Program Systems Engineering and Computer Science at COPPE/UFRJ. He holds a degree in Software Engineering from Federal University of Amazonas (UFAM). His areas of interest are: Software Engineering, Software Ecosystem, Developer Experience, Experimental Software Engineering and Human-Computer Interaction.
Guilherme H. Travassos is a professor of Software Engineering at COPPE/UFRJ and a CNPq (Brazilian Research Council) Researcher. He holds a D.Sc. in Systems Engineering and Computer Science from COPPE/UFRJ, with a postdoc in Experimental Software Engineering at UMCP/USA. He leads the Experimental Software Engineering Group at COPPE/UFRJ and is a member of ISERN, SBC, and ACM. Apart from that, he takes part in the editorial board of Elsevier - IST, World Scientific - IJSEKE and Springer - JSERD. Further information at http://www.cos.ufrj.br/~ght.

Funding

We thank CAPES for the postdoctoral and CNPq for the graduate grants. Professor Travassos is a CNPq Researcher.

Availability of data and materials

As indicated in the paper, most of the synthesis data is available in the Evidence Factory tool. The synthesis presented in this paper can be accessed at http://evidencefactory.lens-ese.cos.ufrj.br/synthesis/editor/80416.
Additional data used in the analysis, which are important for understanding and interpreting the results, were presented in the two appendices at the end of this paper.

Authors' contributions

PSMS, the first author, conceived, planned, and executed the study. ACB and BPS executed the study with the first author and assisted with data handling and tabulation. The last author, GHT, helped with conceiving and planning the study. Besides, he heavily worked in defining the text structure and in its revision. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 1 March 2018 Accepted: 28 September 2018
Published online: 19 October 2018

References

Abrantes JF, Travassos GH (2013) Towards pertinent characteristics of agility and agile practices for software processes. CLEI Electron J 16:6–6
Ahmad MO, Dennehy D, Conboy K, Oivo M (2018) Kanban in software engineering: a systematic mapping study. J Syst Softw 137:96–113. https://doi.org/10.1016/j.jss.2017.11.045
Ahmad MO, Markkula J, Oivo M (2013) Kanban in software development: a systematic literature review. In: 2013 39th Euromicro conference on software engineering and advanced applications, pp 9–16
Al-Baik O, Miller J (2015) The kanban approach, between agility and leanness: a systematic review. Empir Softw Eng 20:1861–1897. https://doi.org/10.1007/s10664-014-9340-x
Anderson DJ (2010) Kanban: successful evolutionary change for your technology business, 3.8.2010. Blue Hole Press, Sequim
Andriole SJ (2017) The death of big software. Commun ACM 60:29–32. https://doi.org/10.1145/3152722
Atkins D, Best D, Briss PA et al (2004) Grading quality of evidence and strength of recommendations. BMJ 328:1490. https://doi.org/10.1136/bmj.328.7454.1490
Auerbach CF, Silverstein LB (2003) Qualitative data: an introduction to coding and analysis. New York University Press, New York
Barney JB (1986) Organizational culture: can it be a source of sustained competitive advantage? Acad Manag Rev 11:656–665
Bloch I (1996) Some aspects of Dempster-Shafer evidence theory for classification of multi-modality medical images taking partial volume effect into account. Pattern Recogn Lett 17:905–919. https://doi.org/10.1016/0167-8655(96)00039-6
Borenstein M, Hedges LV, Higgins JPT, Rothstein HR (2009) Introduction to Meta-analysis, 1 edition. Wiley, Chichester
Britten N, Campbell R, Pope C et al (2002) Using meta ethnography to synthesise qualitative research: a worked example. J Health Serv Res Policy 7:209–215. https://doi.org/10.1258/135581902320432732
Chapetta WA (2016) Proposal for the Definiton of a theoretical model for observing productivity in software processes. Qualifying Examination Proposal, Federal University of Rio de Janeiro
Conboy K (2009) Agility from first principles: reconstructing the concept of agility in information systems development. Inf Syst Res 20:329–354. https://doi.org/10.1287/isre.1090.0236
Corona E, Pani F (2013) A review of lean-Kanban approaches in the software development. WSEAS Trans Inf Sci Appl 10:1–13
Cruzes DS, Dybå T (2011) Research synthesis in software engineering: a tertiary study. Inf Softw Technol 53:440–455. https://doi.org/10.1016/j.infsof.2011.01.004
Da Silva FQB, Cruz SSJO, Gouveia TB, Capretz LF (2013) Using Meta-ethnography to synthesize research: a worked example of the relations between personality and software team processes. In: 2013 ACM / IEEE international symposium on empirical software engineering and measurement. pp 153–162
Dennehy D, Conboy K (2017) Going with the flow: an activity theory analysis of flow techniques in software development. J Syst Softw 133:160–173. https://doi.org/10.1016/j.jss.2016.10.003
DeRemer F, Kron HH (1976) Programming-in-the-large versus programming-in-the-small. IEEE Trans Softw Eng SE-2:80–86. https://doi.org/10.1109/TSE.1976.233534
Fitzgerald B, Musiał M, Stol K-J (2014) Evidence-based decision making in lean software Project Management. In: Companion proceedings of the 36th international conference on software engineering. ACM, New York, pp 93–102
Heikkilä VT, Paasivaara M, Lassenius C (2016) Teaching University students Kanban with a collaborative board game. In: Proceedings of the 38th international conference on software engineering companion. ACM, New York, pp 471–480
ISO (2005) ISO 9000:2005 — quality management systems — fundamentals and vocabulary. International Organization for Standardization, Geneva
ISO (2017a) ISO 25010:2011 — systems and software engineering — systems and software quality requirements and evaluation (SQuaRE) — system and software quality models. International Organization for Standardization, Geneva
ISO (2017b) ISO 12052:2017 — health informatics — digital imaging and communication in medicine (DICOM) including workflow and data management. International Organization for Standardization, Geneva
Kitchenham B, Pfleeger SL (1996) Software quality: the elusive target [special issues section]. IEEE Softw 13:12–21. https://doi.org/10.1109/52.476281
Maglyas A, Nikula U, Smolander K (2012) Lean solutions to software product management problems. IEEE Softw 29:40–46. https://doi.org/10.1109/MS.2012.108
Martinez-Fernandez S, Santos PSM, Ayala CP et al (2015) Aggregating empirical evidence about the benefits and drawbacks of software reference architectures. In: 2015 ACM/IEEE international symposium on empirical software engineering and measurement (ESEM), pp 1–10
Ohno T, Bodek N (1988) Toyota production system: beyond large-scale production, 1st edn. Productivity Press, Cambridge
Pawar KS, Menon U, Riedel JCKH (1994) Time to market. Integr Manuf Syst 5:14–22. https://doi.org/10.1108/09576069410815765

Pickard LM, Kitchenham BA, Jones PW (1998) Combining empirical results in software engineering. Inf Softw Technol 40:811–821. https://doi.org/10.1016/S0950-5849(98)00101-3

Poppendieck M, Cusumano MA (2012) Lean software development: a tutorial. IEEE Softw 29:26–32. https://doi.org/10.1109/MS.2012.107

Poppendieck M, Poppendieck T (2013) The lean mindset: ask the right questions, 1st edn. Addison-Wesley Professional, Upper Saddle River

Santos PSM (2015) Evidence representation and aggregation in software engineering using theoretical structures and belief Fuctions. Thesis, Federal University of Rio de Janeiro

Santos PSM, Travassos GH (2017a) Structured synthesis method: the evidence factory tool. In: 2017 ACM/IEEE international symposium on empirical software engineering and measurement (ESEM), pp 480–481

Santos PSM, Nascimento IE, Travassos GH (2015) A computational infrastructure for research synthesis in software engineering. In: XVIII Ibero-American conference on software engineering, track: XVII workshop on experimental software engineering. Lima: Curran Associates, pp 309–322

Santos PSM, Travassos GH (2013) On the representation and aggregation of evidence in software engineering: a theory and belief-based perspective. Electron Notes Theor Comput Sci 292:95–118. https://doi.org/10.1016/j.entcs.2013.02.008

Santos PSM, Travassos GH (2016) Scientific knowledge engineering: a conceptual delineation and overview of the state of the art. Knowl Eng Rev 31:167–199. https://doi.org/10.1017/S0269888916000011

Santos PSM, Travassos GH (2017b) Evidence of usage-based Reading effects by using the structured synthesis method. Technical Report, Federal University of Rio de Janeiro (UFRJ/COPPE)

Shafer G (1976) A mathematical theory of evidence. Princeton University Press, Princeton

Sjøberg DIK, Dybå T, Anda BCD, Hannay JE (2008) Building theories in software engineering. In: Shull F, Singer J, Sjøberg DIK (eds) Guide to advanced empirical software engineering. Springer London, London, pp 312–336

Staats BR, Brunner DJ, Upton DM (2011) Lean principles, learning, and knowledge work: evidence from a software services provider. J Oper Manag 29:376–390. https://doi.org/10.1016/j.jom.2010.11.005

Stevenson A (2010) Oxford Dictionary of English, 3rd Revised ed. Oxford University Press, New York

Tannenbaum SI (1997) Enhancing continuous learning: diagnostic findings from multiple companies. Hum Resour Manag 36:437–452. https://doi.org/10.1002/(SICI)1099-050X(199724)36:4<437::AID-HRM7>3.0.CO;2-W

Versionone (2017) The 11th annual state of agile survey. https://explore.versionone.com/state-of-agile. Accessed 28 Jan 2018

Womack JP, Jones DT (1997) Lean thinking—banish waste and create wealth in your corporation. J Oper Res Soc 48:1148–1148. https://doi.org/10.1057/palgrave.jors.2600967