The Journal of Systems & Software 178 (2021) 110973

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

# Synthesizing researches on Knowledge Management and Agile Software Development using the Meta-ethnography method☆

Bianca Minetto Napoleão [a], Érica Ferreira de Souza [a,*], Glauco Antonio Ruiz [a], Katia Romero Felizardo [a], Giovani Volnei Meinerz [a], Nandamudi Lankalapalli Vijaykumar [b,c]

[a] Department of Computer, Federal University of Technology, Paraná - UTFPR, Cornélio Procópio, Paraná, Brazil
[b] National Institute for Space Research (INPE), Laboratory of Computing and Applied Mathematics (LAC), São José dos Campos, São Paulo, Brazil
[c] Federal University of São Paulo (Unifesp), Institute of Science & Technology (ICT), São José dos Campos, São Paulo, Brazil

ARTICLE INFO

*Article history:*
Received 1 November 2019
Received in revised form 23 February 2021
Accepted 5 April 2021
Available online 18 April 2021

*Keywords:*
Meta-ethnography
Knowledge management
Agile software development
Scrum
XP

ABSTRACT

**Context:** Software development processes are considered as knowledge intensive and therefore Knowledge Management (KM) can be applied to efficiently manage the knowledge generated. Agile practices can benefit software organizations in terms of KM. Some studies have already presented evidence about this relationship. However, the intersection of these two areas still requires further clarification.
**Objective:** This study synthesizes research on KM and Agile Software Development (ASD) using the meta-ethnography method considering Scrum and XP frameworks.
**Method:** In order to achieve the proposed goal, first, a diagnostic was conducted in different project domains with agile and traditional software development in order to identify the performance of KM activities. Second, the phases of the meta-ethnography analysis method were applied in a set of studies selected from a tertiary review on KM and ASD, as well as classic guides and area references. Finally, the relationships that were identified among the areas investigated were analyzed from interviews with agile development methodology experts.
**Results:** The most common activity investigated between KM and ASD is knowledge sharing. However, in the practical view of software development companies, the attention is on how to use the generated knowledge.
**Conclusion:** The clarification of how KM is present in each agile value, practices, and artifacts allows a reflection on how much knowledge was created, shared, and applied during ASD. Besides, such results presented in this study enable organizations to know each other better and to explore more each KM activity, thus contributing to delivering more value to the customer.


## 1. Introduction

One of the challenges of Software Engineering is creating systematic strategies to integrate knowledge involved in a project. Software organizations seek solutions that emphasize different knowledge types when planning initiatives to manage them (Bjørnson and Dingsøyr, 2008). A systematic routine for capturing knowledge is important in an organization, making knowledge transparent to everyone involved to be benefited from accumulated knowledge (Levy and Hazzan, 2009). However, while the importance of working with knowledge has been widely recognized in many areas, managing that knowledge is still a challenge for software development organizations.

Since software development is a knowledge intensive and collaborative activity, the success of the project depends on the knowledge and experience of the developers (Vasanthapriyan et al., 2015). However, those involved in the project face problems such as: difficulty in systematizing the information generated throughout the software processes; difficulty in reusing knowledge generated from one project in another; loss of the organization's intellectual capital; and employees normally reluctant to share their knowledge (Andrade et al., 2013; Souza et al., 2015). This is because most of the knowledge generated in organizations is not processed and it becomes difficult to articulate (Abdullah et al., 2011; Li and Zhang, 2012; Andrade et al., 2013; Souza et al., 2015). In this respect, Knowledge Management (KM) principles can help organizations to represent and share the knowledge generated in the organization (Davenport and Prusak, 2000).

☆ Editor: Doo-Hwan Bae.

* Corresponding author.
*E-mail addresses:* biancanapoleao@alunos.utfpr.edu.br (B.M. Napoleão), ericasouza@utfpr.edu.br (É.F. de Souza), glaucoruiz@alunos.utfpr.edu.br (G.A. Ruiz), katiascannavino@utfpr.edu.br (K.R. Felizardo), giovanimeinerz@utfpr.edu.br (G.V. Meinerz), vijay.nl@inpe.br, vijaykumar@unifesp.br (N.L. Vijaykumar).

https://doi.org/10.1016/j.jss.2021.110973
0164-1212/

KM promotes creating, sharing, and storing knowledge to make it accessible and reusable within the organization (O'Leary and Studer, 2001). According to O'Leary (1998), KM formally manages knowledge resources to facilitate access and reuse. Nonaka and Takeuchi (1995) use the tacit–explicit distinction to differentiate inarticulate and articulate knowledge. Tacit knowledge is based on experience and usually remains only in people's minds. Tacit knowledge encompasses knowledge associated with senses, skills, experiences, or intuition. Explicit knowledge, in turn, represents the knowledge that can be easily documented and shared as long as it is objective and rational. The creation of knowledge can be considered an interaction between tacit and explicit knowledge. In this interaction, there is a process to transform tacit knowledge into explicit. Since tacit knowledge is highly internalized, this process is important to knowledge sharing (Nonaka and Takeuchi, 1995).

Traditional software development involves using multiple documents to capture and represent knowledge related to the various stages of software development lifecycles (Wendorff and Apshvalka, 1998). In this traditional scenario, explicit knowledge is decisive. On the other hand, Agile Software Development (ASD) is considered a lightweight software development. Agile frameworks emphasize collaboration among team members in applying and sharing knowledge. Agile frameworks prioritize tacit knowledge, encouraging individual, team, and customer communications and interactions (Andriyani et al., 2017). One of the main challenges faced by software development organizations working with ASD is to transform the tacit knowledge into explicit (e.g., sprint planning or sprint backlog). Once the externalization of tacit to explicit knowledge occurs, it becomes easier to be shared and communicated (Nonaka and Takeuchi, 1995).

Agile practices and KM present common activities that can benefit software organizations to promote knowledge sharing, team communication, knowledge reuse, and the collaborative process. However, according to Cabral et al. (2014), there is still a gap about what emerges from the intersection of these two areas requiring further clarification. For this reason, KM in ASD has been treated as a broad research topic, resulting in various relationships between its schools and concepts.

Based on the above context, we intend to understand how ASD is inherent to KM activities. We decided to explore the following Research Question (RQ) to guide us to evidence the relations in both these areas:

*How do agile values and practices relate to knowledge management activities?*

First, a KM diagnostic was conducted in different project domains with agile and traditional developments in order to evaluate KM activities present in the organization. A KM diagnostic analyzes an organization's current state on KM and can help its members understand their real needs before devoting costly efforts to KM implementation and thus better target KM application initiatives at strategic points (Bukowitz and Williams, 2000). Conducting KM diagnostic in different domains of software development shows how KM activities are present in environments with agile or traditional practices.

Once KM is measured employing a diagnostic in order to answer the RQ of this study, researches on KM and ASD were synthesized using the meta-ethnographic method (Noblit and Hare, 1988). Meta-ethnography is an interpretive method to synthesize research by absorbing the concepts identified in the relevant studies into a higher-order theoretical structure (Noblit and Hare, 1988; Fu et al., 2019). A meta-ethnography should not attempt to produce gross generalizations across studies with disparate goals and from too distinct contexts (Noblit and Hare, 1988). Therefore, aggregate methods, such as meta-analysis or integrative reviews, that synthesize from an exhaustive list of studies, are unable to increase external validity on the results. The ethnographic method requires several readings of an initial set of studies, data extraction, consensus gathering, reaching agreements, and verifying possible inconsistencies in interpretations. It also depends on team maturity that conducts research.

This research is an extension of a preliminary study that synthesized Scrum framework events and artifacts with KM's main activities (Ruiz et al., 2018). Scrum is an agile framework for developing, delivering, and maintaining software products (Schwaber and Sutherland, 2017). In this initial study, the seven phases of meta-ethnographic method were applied in five studies selected from a review of secondary studies in KM and ASD (Neves et al., 2011; Cabral et al., 2014; Paredes et al., 2014; Borrego et al., 2017; Andriyani et al., 2017). Main concepts and relationships were able to be identified between the main KM activities (create, share, and apply the knowledge) with the Scrum framework artifacts and activities. In addition, the concepts and relationships identified between the areas investigated were analyzed based on interviews with three ASD professionals.

This extended research presents both advances of the results already identified in Ruiz et al. (2018), and incorporation of new practices for KM diagnostic. The main activities conducted in this study are: (i) conducting a KM diagnostic process in software organizations that maintain traditional and agile software product development practices; (ii) updating the review conducted in Ruiz et al. (2018) which identified new studies to be part of the initial set of studies to synthesize the investigated areas; (iii) updating and conducting the synthesis process considering Scrum and Extreme Programming (XP) agile software development frameworks; and (iv) conducting interviews with professionals who present a practical view on ASD in order to collect opinions about the identified relationships between KM and ASD.

The remainder of this study is structured as follows. Section 2 reviews the literature of KM, agile development, and the Meta-ethnography method. Section 3 presents the related work. Section 4 formally introduces the methods and procedures used to conduct the research, as well as the main results. Section 5 discusses and highlights some points of our research. Lastly, conclusions, remarks and future directions are described in Section 6.

## 2. Background

In this section, the main concepts of this study are discussed.

### 2.1. Knowledge management

KM is the process through which organizations generate value from their intellectual assets (tacit and explicit knowledge) (Bukowitz and Williams, 2000). Most often, generating value from such assets involves sharing them among employees, departments, and even with other companies to devise best practices.

KM can also be defined as a set of organizational activities that must be performed systematically. Effective KM requires an organization to execute activities such as to identify, generate, acquire, diffuse, and capture knowledge. The way how activities are organized is called KM cycles (also known as KM models) (Dalkir, 2005).

KM cycles can be used to guide how to conduct KM activities. KM cycles have an objective to help to identify and locate knowledge and knowledge sources within the organization. According to Dalkir (2005), the four most known KM cycles are presented by Wiig (1993), Meyer and Zack (1996), Levy and Hazzan (1999)

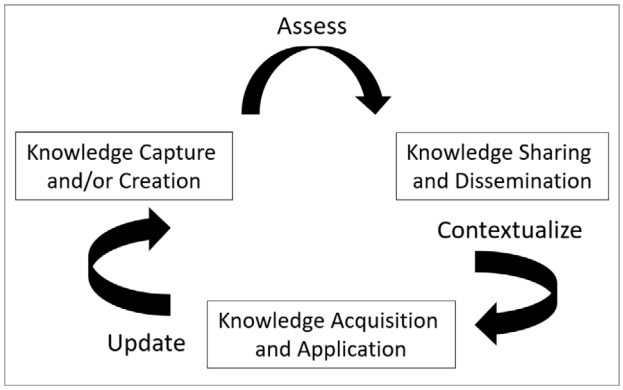


**Fig. 1.** An integrated KM Cycle. *Source:* Adapted from Dalkir (2005).

and Bukowitz and Williams (2000). These KM cycles identify and locate knowledge and knowledge sources within the organization from several activities. However, Dalkir (2005) also mentions that a similarity lack exists concerning the terms used to describe the major activities in the KM cycle. On the basis of main KM cycles Dalkir (2005), one can distill an integrated KM cycle represented by interaction of three activities:

- **Knowledge capture and/or creation:** Knowledge capture refers to the identification of existing internal and/or external knowledge from the environment. Knowledge creation is the development of new knowledge that did not have a previous existence within the organization. In this activity, tacit knowledge is captured or elicited, and explicit knowledge is organized or coded.
- **Knowledge sharing and dissemination:** Once knowledge has been captured and coded, it needs to be shared and disseminated throughout the organization. Several practices can be used to share knowledge within the organization, such as team meetings, written instructions, ad-hoc information, verbally disseminated information, intranet posts or video clips.
- **Knowledge acquisition and application:** Knowledge that has been captured, coded, shared, and otherwise made available is put to use. KM can succeed only if the knowledge is used. However, it now becomes imperative to understand which knowledge is of use to which set of people and how best to make it available. The use of KM systems, such as expertise location systems or content management systems can be designed to optimize knowledge application on an organization-wide basis.

The transition between the three activities is illustrated in Fig. 1. In the integrated KM cycle, the transition from knowledge capture/creation to knowledge sharing and dissemination, knowledge content is assessed. Knowledge is then contextualized in order to be understood (“acquisition”) and used (“application”). This process then feeds back into the first activity in order to update the knowledge content (Dalkir, 2005).

In this study, the integrated KM cycle presented in Dalkir (2005), was used to represent the main KM activities serving as input for the synthesis method presented in Section 2.5.

### 2.2. Knowledge management and software engineering

One of the main characteristics of software engineering is the high volume of information that is generated and manipulated in the organization. Integrating KM with software engineering has brought much discussion about how to manage knowledge in the organization. According to Bjørnson and Dingsøyr (2008), research related to KM and software engineering focuses, especially, on ways and means to share knowledge.

Knowledge in a software engineering organization should be properly captured, stored, and reused when it is needed. KM principles and techniques application aims at facilitating knowledge flow and utilization across the software engineering process. In this environment knowledge needs to be updated all the time since software development environment technologies used are often changing (Vasanthapriyan et al., 2015).

In software engineering, there is a lot of ongoing discussion about how to manage knowledge, or how to promote learning in the organization. When organizations talk about KM, one of the main practices is to develop an “Experience Factory” (Basili et al., 1994). In a software development environment, the experience of each activity conducted can be collected, packed, and stored in a knowledge base to be easily reused, documented, and, therefore, accessed by several members of the organization. The explicit knowledge can be described by drawings and writings, consequently, it can be easily used and shared (Bjørnson and Dingsøyr, 2008).

Vasanthapriyan et al. (2015) describe that effective management of the software development process became a need for the software development organizations to survive in the competitive scenario. In order to get strengths in the development process business, the organizations need to execute the software development efficiently, introducing KM principles that support the software process. Therefore, introducing KM in software engineering concepts is essential. Beyond that, KM helps the organization to improve the decision-making process, improving innovation and organization performance, and also helping the organization to sustain the competitiveness.

KM principles are present in software development organizations, which use traditional practices as well as those using agile practices. Regarding those with agile practices, lots of KM activities can be considered to bear a resemblance, for example, communication and knowledge sharing (Cabral et al., 2014). The study about KM activities in agile practices might help development teams to improve learning and to collaborate with the organization’s knowledge evolution, leading to high value deliveries and, consequently, raising customer satisfaction.

### 2.3. Agile software development

The traditional software development process can limit developers since it sometimes turns into a complex and expensive process. Besides, it is extremely focused on documentation. This fact highlights the emergence of agile methodologies (Schön et al., 2015). The projects using agile methodologies assume that change is common in software projects and thus the ongoing planning is considered to be quite valuable, emphasizing human aspects and adaptability to rapid changes.

In 2001, leaders of different streams joined hands and created the Manifesto for ASD (Beck et al., 2001). Agile Manifesto includes values and principles that help to optimize the software development process and also have a strong influence on present practices of team collaboration within ASD (Schön et al., 2015). The Agile Manifesto provides four core values:

(I) Individuals and interactions over processes and tools;
(II) Working software over comprehensive documentation;
(III) Customer collaboration over contract negotiation; and
(IV) Responding to change following a plan.

In order to represent the relation of agile with KM, in this study we use Scrum and Extreme Programming (XP) frameworks. We chose Scrum and XP as input to the synthesis method in this study since they are the most mentioned frameworks in the eight studies that we considered in the meta-ethnography conduction in this study (see Section 4.2). Also, Scrum is considered the most commonly used framework for software development Farrukh and Tariq (2017), and XP practices have been used in the customization of new frameworks, for example, Scrum/XP Hybrid (Farrukh and Tariq, 2017; CollabNet VersionOne, 2019).

#### 2.3.1. Scrum

Schwaber and Sutherland (2017) official document defines Scrum as a framework for developing, delivering, and sustaining complex products. According to Sutherland and Schwaber (2012), the guide focuses on two core elements: Scrum events (more precisely in Sprint event, which is a container for all other events) and Scrum artifacts. These elements are briefly presented below.

**Scrum Events.** Prescribed events are used in Scrum to create regularity and to minimize the need for meetings that were not scheduled. The events are specifically designed to enable critical transparency and inspection. Each event in Scrum is a formal opportunity to inspect and adapt something. One of the main events in Scrum is Sprint. Sprint is a process that lasts on average one month to deliver an incremented version. Sprints contain and consist of the Sprint Planning, Daily Scrums, the development work, the Sprint Review, and the Sprint Retrospective (Sutherland and Schwaber, 2012; Schwaber and Sutherland, 2017).

The work to be done at Sprint is defined in the Sprint planning meeting. This planning is done by the entire Scrum team collaboratively. Daily Scrum, Scrum's daily meetings, are meetings that last fifteen minutes at most for the development teams to synchronize activities and build up a plan for the next 24 h. According to Pressman (2014), during the meeting, each member of the development team clarifies the following questions: (i) What have you done since the last team meeting?; (ii) What obstacles are you facing?; and (iii) What do you plan to do until the next team meeting?

The Sprint Review Meeting is performed after the Sprint to verify the increase and to adapt to the backlog of the product if necessary. The Sprint Retrospective is an opportunity for the scrum team to inspect and create a plan for improvements that should be valid for the next Sprint (Sutherland and Schwaber, 2012).

**Scrum Artifacts.** Scrum artifacts differ from common artifacts because they are designed to provide transparency and enable inspection and adaptation ensuring that everyone has the same understanding about a particular artifact. Product Backlog is an orderly list of everything that can be necessary for the product and it is the only source for the requirements of any needed changes to be made in the product. The Product Backlog lists all features, every functionality, functions, requirements, improvements, and repairs that represent changes to be made for the next version of the product. Its items have attributes descriptions, order, and estimation of effort. Generally, it is ordered by the value it aggregates to the business, in terms of risk, priority, and need. The very first items of the Product Backlog determine more immediate development activities (Sutherland and Schwaber, 2012; Schwaber and Sutherland, 2017).

Sprint Backlog is a set of the Product Backlog items selected for a Sprint and for a plan to obtain a product increase and to achieve the Sprint goals. Sprint Backlog defines the work to be done by the development team in order to make the Product Backlog items into 'Ready' increases (Sutherland and Schwaber, 2012). At the end of a Sprint, the new increment has to be 'Ready', which means that it is in a usable condition and meets the definition 'Ready' of the Scrum Team.

The scrum team is composed of the Product Owner, the Development Team, and the Scrum Master. These teams are self-organized, and they choose the best way to finalize the work. The teams are also cross-functional and multi-functional and all of them have the necessary skills to carry out the work without dependence on others who are not part of the team.

#### 2.3.2. Extreme programming (XP)

XP is a style of software development focusing on the application of programming techniques, clear communication, and teamwork that allows software development based on the values of communication, feedback, simplicity, courage, and respect (Beck and Andres, 2004). XP employs an object-oriented methodology as its development paradigm and contains rules and practices in the context of four methodological activities: planning, design, codification, and testing (Pressman, 2014).

The planning activity consists of creating a set of stories from users who describe a set of characteristics and functionalities required for the software to be developed. Both the customers and developers work together to decide how to group up stories for the next release. After the first release is finished, the XP team calculates the project speed, which means the number of stories implemented in the first version. With this, it is possible to estimate the delivery time and schedule the following versions. As the project moves, the customer can add new stories (Beck and Andres, 2004; Pressman, 2014).

A simple project is desirable over a complex project. Thus, the XP follows the KIS principle (keep it simple) strictly. The project provides implementation guidelines to a story as it is written, nothing more, nothing less. XP encourages the use of CRC cards (Class–Responsibility–Collaborator) that identify and organize the stories. The cards are pinned on a prominent wall (Beck and Andres, 2004). XP also encourages refactoring, which consists of an enhancement of the code after it is written (Pressman, 2014).

A key-concept during the codification activity at XP is programming in pairs. Pair programming is a dialog between two people simultaneously programming the story code (analyzing, designing and testing) (Beck and Andres, 2004).

It is important to emphasize the collective ownership of the source code in XP. The source code of a program has no exclusive owner and there is no need for formal permission to make any modifications. Even though control tools are necessary, so that work is not lost, this XP practice allows the entire team to know every part of the system (Pressman, 2014).

Another point that deserves to be highlighted is continuous integration. Each integration or functionality made for the software must be integrated immediately to the current version of the system, thus, minimizing conflict problems among older versions and enabling the awareness of the real development status (Beck and Andres, 2004).

XP framework recommends creating a series of unitary tests, before coding, which will exercise every single story that must be included in the actual version. This is because once those tests are created, the developer knows what is needed to be done to pass the unitary test. These unitary tests are created in a way that they can be automated and easy to be executed because they will be executed over and over again. Finally, the acceptance tests, or customer tests, are used by the customers based on their stories, focusing on the characteristics and functionalities of the global system that are visible and subject to change (Beck and Andres, 2004; Pressman, 2014).

### 2.4. Knowledge management

Several activities promote KM in an organization, such as creating, sharing, and reusing the knowledge. It is also possible to find in the literature KM cycles that allow us to follow

a systematic process of how to promote such activities in the organization (Dalkir, 2005). However, while there are several KM approaches, conducting a diagnostic in an organization can identify which knowledge activities need, in fact, improvement or present the best cost-effectiveness for the organization in terms of KM (Bukowitz and Williams, 2000; Rodriguez-Elias et al., 2008).

Analyzing the organization's current state concerning KM activities can help the organizations understand their actual needs before dedicating expensive efforts in the KM implementation and thus better target the KM application initiatives in companies' strategic points. Therefore, a knowledge diagnostic will help the organization to determine which knowledge is being managed and how well it is managed. Knowledge audit helps to make the knowledge in the company visible (Nejati, 2010).

KM has been introduced as an effective process to monitor the performance of KM practices in an organization (Nejati, 2010). Usually, the diagnostic is conducted through some type of research instrument, most of the time with questionnaires or interviews being used (Esteves, 2017). The issues are measured, and the results show where the organization can guide its KM efforts. Some known models (or cycles) in the literature for KM diagnostic are *Asian Productivity Organization* (APO)[1], *Organizational Knowledge Assessment* (OKA) (Fonseca, 2006), Seven Dimensions of Knowledge Management (Terra, 2001) and Knowledge Management (KMD) (Bukowitz and Williams, 2000).

### 2.5. Meta-ethnography method

Noblit and Hare (1988) define meta-ethnography as an intensive synthesis method involving observation, interviewing, and document review. It enables a systematic and detailed understanding of how studies are related. This relation is done through the comparison of findings within and across studies. Meta-ethnography is the method of qualitative synthesis most widely used in health and education research. According to Noblit and Hare (1988), a meta-ethnography could be identified as a study rather than a method since it helps a researcher to build an interpretative rather than an aggregate description of the evidence being synthesized.

In summary, researchers select, analyze and interpret qualitative studies through a process of translation, in which the studies are coded into metaphors, which provide an interpretation of the entire topic, in order to answer focused questions on a specific topic and gain new insights.

Meta-ethnography is often confused with other qualitative methods (synthesis) because of the similarity between them (Fu et al., 2019). Examples of other qualitative methods are Thematic Synthesis and Grounded Theory. The difference is that Thematic Synthesis is an extension of thematic analysis, which is often used to identify, analyze, and report patterns within data in primary research, and further creates a model of higher-order themes with descriptive summaries (Cruzes and Dybå, 2011a). On the other hand, meta-ethnography provides a new interpretation that goes beyond all the primary or a picture of the whole phenomenon under study from studies of its parts. The grounded theory further generates a new theory that can explain the targeted phenomenon using the data from primary studies (Hannes and Lockwood, 2011).

[1] http://www.apo-tokyo.org/.

## 3. Related work

In Fu et al. (2019) and Cruzes and Dybå (2011b), tertiary studies were conducted on meta-ethnography used in Systematic Literature Reviews (SLRs) in Software Engineering. Both studies present several benefits of meta-ethnography, but also highlight a general lack of attention paid to research synthesis in software engineering. The authors reported that there were only a few SLRs that employed the meta-ethnography method.

Fu et al. (2019), for example, mention that SLRs that appear to have used meta-ethnography for data synthesis have used some featured techniques of meta-ethnography rather than the complete defined method consisting of seven steps. In addition, Fu et al. (2019) mention that in some reviews the authors did not explicitly state the use of meta-ethnography or did not even know whether they were using it. There are some relationships between SLR and meta-ethnography. Hence, some authors use SLR to identify the main evidence on a particular research topic, and then they use meta-ethnography to identify the relationships and connections among the evidence in Data Extraction and Data Synthesis activities of the SLR.

As our study aims to synthesize what has been investigated between two distinct areas through meta-ethnography considering KM, we present below the only work found so far that explicitly presents a synthesis of KM with other areas using the seven steps of meta-ethnography.

Da Silva et al. (2013) explore the use of meta-ethnography in the synthesis of empirical studies in KM through an example using studies on the relationships between personality and software team processes. Thus, the seven phases of meta-ethnography were applied in a set of studies selected from a systematic review previously developed to evaluate the adequacy of meta-ethnography in this domain with respect to the ease of use, utility, and reliability of the results.

Common concepts were identified through reading and interpretation of the studies. Then, second order translations have been built and used to synthesize a relation model between the software team's personality and processes (Da Silva et al., 2013). Meta-ethnography is adequate in the synthesis of empirical studies, even in the context of studies of mixed methods. Da Silva et al. (2013) also argue that the researcher who is dedicated to the use of meta-ethnography should be aware that the set of studies to synthesize can greatly influence the consistency and the reliability of the resulting synthesis.

## 4. Synthesizing researches on knowledge management and agile software development

The goal of this study is to understand how ASD is inherent to KM activities. In order to achieve this goal, we combined a KM with the application of the meta-ethnography method. Fig. 2 illustrates the research synthesis process followed during the conduction of this study.

As can be observed in Fig. 2, first we conducted a KM in different project domains with agile and traditional software development to identify the performance of KM activities in practice. Next, we applied the meta-ethnography method considering a set of studies from an updated tertiary study. Steps 2, 3, 4, and 5 of the meta-ethnography method used the tertiary study results as a basis for their conduction. Also, step 5 considered classic guides and references (Beck et al. (2001), Schwaber and Sutherland (2017)) to translate Scrum and XP frameworks. Finally, during the synthesis of translations, we collected opinions and/or impressions about the translations through interviews with ASD experts. Details regarding the diagnostic and the meta-ethnography method executed are presented in Sections 4.1 and 4.2, respectively.

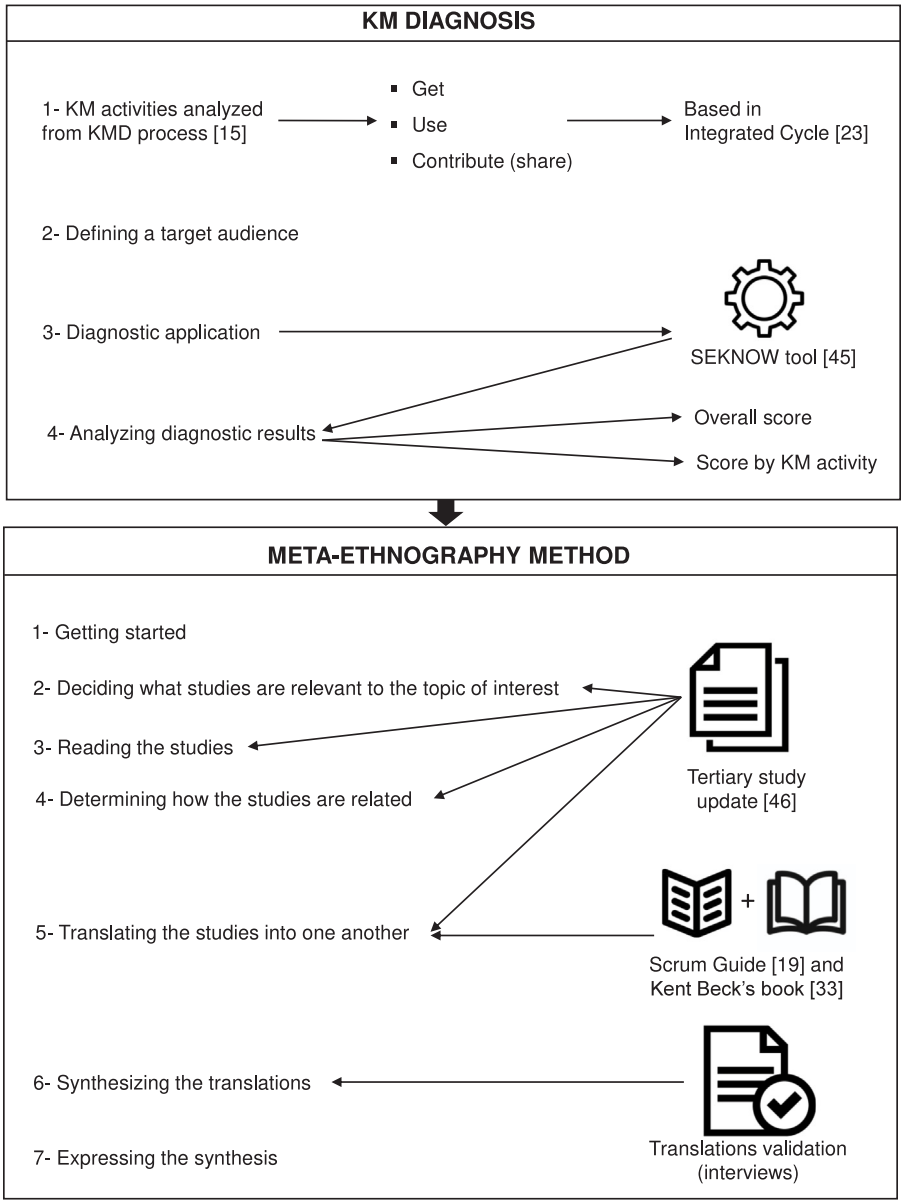


**Fig. 2.** Research synthesis on KM and ASD process.

### 4.1. Diagnostic in different software development approaches

As the first objective of this research, a KM was conducted in different project domains, in order to evaluate KM activities, present in the software organization. It is worth mentioning that the conduction of this diagnostic still presents preliminary results. Even so, conducting this diagnostic along with the obtained results already made us understand how KM activities are present in agile and traditional software development.

We used the KMD process, proposed by Bukowitz and Williams (2000), in order to conduct the diagnostic. Although there are other KM processes in the literature, as previously presented, the KMD process allows to measure several KM activities in an organization (Esteves, 2017; Maciel, 2019). The objective of our study is to measure only KM activities, so the KMD process proved to be directly aligned with that objective. KMD has an analysis of seven KM activities, also called sections by the authors Bukowitz and Williams (2000): get, use, learn, contribute (share), assess, build/sustain and discard.

KMD process enables the analysis of KM activities in organizations considering a performance score. Each activity contains a list of 20 statements that are evaluated by the organization. At the end of the evaluation, a score is defined. The score is defined either by KM activity or by an overall score considering all activities.

The organization must decide to what degree the statement describes it ("Strong (S)", "Moderate (M)", "Weak (W)"). The higher the score obtained, the better the performance of that activity analyzed. In order to compute the KM activity score, the organization should use the following formula (Table 1):

**Table 1**
Calculation of score by activity Bukowitz and Williams (2000).

| | | | |
|---|---|---|---|
| Number of S | [ ] | x 3 = | $X$ |
| Number of M | [ ] | x 2 = | $Y$ |
| Number of W | [ ] | x 1 = | $Z$ |
| Point Score: | | | $X + Y + Z$ |
| Total point score possible 60 | | | |
| Percent Score obtained (point score divided by 60) | | | $\% = R$ |

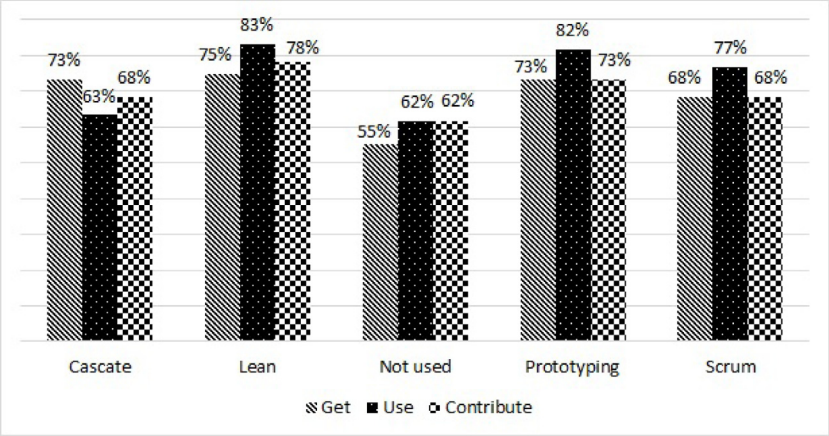


**Fig. 3.** Performance score by KM activity and development approach.

The calculation of the overall score is the sum of all scores (*R*) obtained in the seven activities divided by 420 (60 points x 7 sections). According to Bukowitz and Williams (2000), the average score for all activities (overall score) considered acceptable is 55% and the score per activity is 30% to 70%.

Considering that there is a large number of activities and statements to be analyzed in the KMD process and the organization's effort to respond to it, in this project we defined that the analysis would be conducted in three KMD activities: get, use and contribute (share). These activities were chosen based on Dalkir (2005) definition of integrated cycle, presented in Section 2.1: Capture/Creation (get), Sharing/Dissemination (contribute) and Acquisition/Application (use) (Fig. 2 — KM step 1).

The target audience chosen to participate in the research comprises professionals from Information Technology (IT) who work in software development companies (Fig. 2 — KM step 2). The contact was direct with an organization representative, such as the project manager or the systems analyst. A total of 13 software development organizations participated in the study and applied the diagnosis. 34 employees representing the 13 organizations responded to the diagnostic (average of 3 employees per company). For privacy reasons, organizations names are not mentioned and some of their characteristics are: located in Brazil, small to medium sized software organizations, their main products are specific systems in the area of tax, metrology, games, and also customized systems aimed at meeting the needs of customers from various segments, such as aeronautical, pharmaceutical and automotive.

Out of the 13 organizations, 10 work with agile methods (Scrum and Lean), 2 work with traditional methods (Cascade and Prototyping), and one organization uses neither approach, i.e., development is done in an ad-hoc manner without following any formal approach or a defined development process. One possible justification for it is that the company is small with very few employees.

In order to automate the application of diagnostic and analysis of the results, we developed a tool called Software Engineering KNOWledge management diagnostic (SEKNOW) (Santos et al., 2019). SEKNOW tool has been made available to software development companies to enable faster application of KMD process and reporting results for analysis (Fig. 2 — KM step 3).

As mentioned before, the score obtained by the companies went through a twofold analysis; (i) overall score; and (ii) KM activity (Fig. 2 — KM step 4). The first analysis was concerning an average score of all the KM activities (overall score). Organizations that work with traditional methods obtained an overall score of 73% and companies that work with agile methods obtained a slightly better score of 74%. The organization that does not employ processes or methods in software development presented an overall score of 59%, below the others that employ agile or traditional methods. The organizations that use a process in the software development, either agile or traditional, have greater potential for working with KM activities. However, although 59% score is a lower score, it still draws attention because it presents a relatively good average to consider applying KM activities and within the scoring margin proposed in Bukowitz and Williams (2000).

For each approach used by the organizations, the overall scores presented were: Cascade (68%), Scrum method (72%), Prototyping (77%), and Lean method (79%). Note that when the organization uses a systematic approach to software development, KM-related activities have a higher score. In this context, it is interesting to highlight the high performance presented by companies that use prototyping and agile methods. Prototyping is important because it presents greater communication with the customer since the solution designed is always verified with the real needs of the customer, ensuring the alignment of the demands and their conformance with the presented solutions. Thus, it is possible to minimize the risks by allowing the customer to validate and perform all checks before development. ASD emphasizes collaboration between team members and it prioritizes tacit knowledge, encouraging individual, team, and customer communications (Andriyani et al., 2017). Therefore, agile practices and KM present common activities that can benefit software organizations to promote some KM activities (Ruiz et al., 2018).

Fig. 3 shows the performance score by KM activity and by each software development approach used in the organization. The Lean method and Prototyping presented a higher performance in the three KM activities diagnosed. Contrary to prototyping and agile methods, organizations using the cascade process performed slightly lower than other approaches.

Regarding the performance by KM activity employed by the organization, Fig. 4 presents the scores obtained. The "Use" activity performs a little better (71% average), regardless of the approach used. In the organization's employees' view, the use of knowledge is present in the organization's daily activities, i.e., practices using knowledge are applied more efficiently so that knowledge is well used and disseminated among employees. Some of the practices to use or disseminate the knowledge mentioned by the organizations were daily meetings, Wiki, e-learning, and knowledge base.

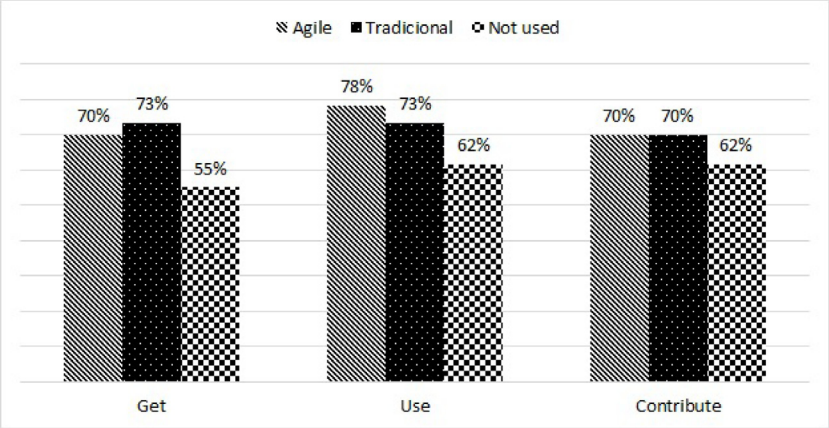


**Fig. 4.** Performance by KM activity.

**Table 2**
Keywords of the search string of the tertiary study in KM and agile development.

| Areas | Keywords |
|---|---|
| Agile | "agile", "extreme programming", "xp method", "scrum", "crystal", "dsdm", "fdd", "feature driven development", "lean software development" |
| KM | "Knowledge Management", "knowledge sharing", "knowledge transfer", "knowledge extraction", "Knowledge discovery", "useful knowledge", "tacit knowledge", "explicit knowledge", "knowledge creation", "knowledge acquisition", "knowledge retention", "knowledge evaluation", "knowledge application", "organization knowledge", "knowledge engineering", "Knowledge representation") |
| Review | "Systematic Literature Review", "Systematic Review", "Systematic Mapping", "Mapping Study", "Mapping Studies", "Systematic Literature Mapping", "Literature Review", "literature analysis" |
| Search string: ("agile" OR "extreme programming" OR "xp method" OR "scrum" OR "crystal" OR "dsdm" OR "fdd" OR "feature driven development" OR "lean software development") AND ("Knowledge Management" OR "knowledge sharing" OR "knowledge transfer" OR "knowledge extraction" OR "Knowledge discovery" OR "useful knowledge" OR "tacit knowledge" OR "explicit knowledge" OR "knowledge creation" OR "knowledge acquisition" OR "knowledge retention" OR "knowledge evaluation" OR "knowledge application" OR "organization knowledge" OR "knowledge engineering" OR "Knowledge representation") AND ("Systematic Literature Review" OR "Systematic Review" OR "Systematic Mapping" OR "Mapping Study" OR "Mapping Studies" OR "Systematic Literature Mapping" OR "Literature Review" OR "Literature Analysis"). | |

### 4.2. Application of meta-ethnography method

The meta-ethnography method is composed of seven stages (Noblit and Hare, 1988), briefly described next:

1. Getting started — the starting point involves identifying a topic that qualitative research might inform. It is relevant to find a topic that could be of interest to both researchers and also for practitioners. Researchers can define a research question to represent the topic and guide the meta-ethnography application.
2. Deciding what studies are relevant to the topic of interest — the goal of the second stage is to find and select studies that are relevant to the topic of interest. It involves activities such as: searching candidate studies; making decisions on inclusion/exclusion; and quality assessment.
3. Reading the studies — during this stage researchers carefully read the set of included studies becoming as familiar as possible with their general content and detail. Researchers can also incorporate data extraction.
4. Determining how the studies are related — initially, researchers can list the key metaphors of each study, be phrases, ideas and/or concepts. Next, the researchers can create a list of these metaphors, juxtaposing them and determining how they are related. Grids or tables can also be used to display metaphors across all studies.
5. Translating the studies into one another — in this stage the goal is to describe how to compare metaphors in one study with those in another. The original method does not define exactly how to do this. One suggestion is to compare the metaphors from study 1 with study 2, and the synthesis of these two studies with study 3, and so on Silva et al. (2013).
6. Synthesizing the translations — when the previous stage results in many metaphors, these can be compared to see if there are common types or if some metaphors can encompass others. Besides, the findings of the synthesis can be represented as diagrams or figures.
7. Expressing the synthesis — the findings of the synthesis can be finally disseminated to interested parties.

The application results of each step of the method in the context of ASD and KM are presented below.

(1) ***Getting started***

In this study, we intend to understand how agile values and practices are inherent to KM activities. We want to perform a synthesis about the intersection of these two areas that needed further clarification for researchers and practitioners interested in the study of ASD and KM. Regarding ASD, we focus on Scrum and XP frameworks. For Scrum, we decided to refine and explore the events and artifacts. In the case of XP, our focus was the XP practices.

(2) ***Deciding what is relevant to the initial interest***

In order to decide which studies are relevant to the initial interest, we conducted a tertiary study looking for secondary studies investigating the state of the art in KM and agile development. Tertiary studies are considered as a review that focuses only on secondary studies, i.e., it is a review about other secondary studies (Kitchenham and Charters, 2007). An early version of this tertiary review was published in Ruiz et al. (2018). The first version was conducted until December 2017 and five studies were identified to compose the initial set to conduct the research

on synthesizing the areas. For this study, an update of this tertiary review, following the guidelines proposed by Kitchenham and Charters (2007), was conducted considering published evidence until April 2019. As a result, three new studies were identified, totaling eight studies for this analysis.

We used the search string shown in Table 2, and was applied in three metadata fields: title, abstract, and keywords. The search string had been syntactically adapted to conform to the particularities of each database source.

The following database sources were used in this tertiary review: Scopus, IEEE Xplore Digital Library, ACM Digital Library, Science Direct, and Engineering Index Compendex. The selection criteria are organized into one inclusion criterion (IC) and five exclusion criteria (EC). The inclusion criterion is: (IC1) The study discusses KM and agile software development methodology. The exclusion criteria are: (EC1) The study is just published as an abstract; (EC2) The study is not written in English; (EC3) The study is an older version of another study already considered; and (EC4) The study is not a secondary study, such as primary study or editorials, summaries of keynotes, workshops, and tutorials.

A total of 205 studies were identified during the search process. First, we eliminated duplicated studies (publications that appear in more than one source), leaving 186 publications. Out of these 186, we selected secondary studies by reading their titles and abstracts and applying the inclusion and exclusion criteria. As a result, a total of 25 studies were selected. Next, the selection criteria were applied considering the full text. A total of eight studies were returned in this stage. Over these eight studies considered relevant, we performed backward snowballing. Snowballing was applied in order to identify additional relevant studies through the reference lists of the eight studies. However, no relevant study was identified. Thus, our initial set of studies was composed of eight relevant studies from the four sources that we searched, presented in Table 3.

In Table 3 the study reference, year of publication, how many primary studies were included in each study, and the title are shown. It is worth mentioning that although our initial set is eight secondary studies, this set increased considering the primary studies returned in each one. 238 primary studies were returned and after removing the duplicate studies, a total of 212 studies primaries remained, which were analyzed when necessary.

(3) ***Reading the studies***

Firstly, each author read individually to identify the relations between ‘‘agile values and KM’’, ‘‘Scrum and KM’’, and ‘‘XP and KM’’. Subsequently, we conducted several meetings in groups to resolve in consensus the divergences. Among the six authors who were conducting this research, two also work in the software industry and they had practical experiences in agile frameworks. Therefore, there were different views: researchers versus industry professionals. Some of the disagreements and discussions that arose in the meetings were that not everything that is proposed in Scrum or XP is in fact applied. Some small adaptations are made in the companies. Thus, it was necessary to reach a consensus of practice versus academy as the initial set of studies was analyzed.

Analyzing the eight selected studies over the years, the intention to map and begin to correlate KM initiatives in ASD is recent, starting basically in 2011, as Table 3 suggests.

Regarding the scope of each study, although all studies investigate KM and ASD, each study has a different purpose. In Paredes et al. (2014), the purpose is to summarize information visualization techniques used during the design and development steps of the software cycle by agile teams. The suggestion is that visualization techniques are important in ASD because they lead to understanding, collaboration, and self-organization. Visualization techniques help Agile teams to increase knowledge sharing when designing, developing, communicating and tracking progress.

Cabral et al. (2014) investigated the major KM concepts and findings in ASD projects. The study focuses mainly on the main outstanding issues in the software development cycle when tacit knowledge’s use is prioritized over explicit knowledge. The authors discussed the main findings from the perspective of a paradigm shift that prioritizes the use of tacit knowledge rather than explicit knowledge.

Andriyani et al. (2017) conducted a systematic review looking for specific agile practices to support KM, the inherent knowledge involved in these agile practices, and how the agile teams manage that knowledge. According to the authors, the most important contribution the review provided was an understanding of KM in ASD that involves managing three different types of knowledge – process, project, and product – by implementing three KM strategies – discussions, artifacts, and visualizations – during agile practices.

The main purpose in Borrego et al. (2017) is architectural KM in Agile and Global Software Development (AGSD). A mapping study was conducted where the authors identified and described approaches to manage architectural knowledge in AGSD teams. These approaches were grouped as artifact-based, communic - ation-based, and methodological-based documentation.

Neves et al. (2011) analyzed and evaluated the knowledge creation and sharing experiences of teams in the ASD domain. A method was developed to evaluate the advantages and limitations of agile practices in knowledge creation and sharing for agile teams. Considering SWOT (Strengths, Weaknesses, Opportunities, Threats) analysis, the authors assessed agile processes, their relationships with knowledge transfer management, and their effects on the productivity of software development teams.

In Indumini and Vasanthapriyan (2018), a mapping study was conducted to identify existing researches on KM initiatives in ASD in order to present the state-of-the-art in the area. Indumini and Vasanthapriyan (2018) summarize some understandings from mapping: (i) the major problem in organizations is low reuse rate of knowledge and barriers in knowledge transfer in ASD is recent research; (ii) Knowledge types which are used in ASD are not identified correctly in the organizations; (iii) reuse of development knowledge is the main purpose of applying KM in ASD; (iv) there is a great concern with explicit knowledge using in ASD; and (v) advanced technologies are used to provide KM in ASD.

Al Hafidz and Sensuse (2019) conducted a Systematic Literature Review to understand more about research development in the KM improvisation in ASD by collecting various themes of improved area and method used. The research question investigated in Al Hafidz and Sensuse (2019) are ‘‘Why improvement in agile still needed?’’ and ‘‘What are the themes of improvement in agile KM proposed by previous research?’’. The following are the reasons for the need for agile improvement: artifact agile lack of documentation detail; communication and information dissemination problems in the organization; and the increasing of agile process complexion so it needs more supporting tools. Several improvements have been done in different categories such as agile documentation, requirement, process, tools, estimation, and decision making. The authors mention that many problems in agile already have had solutions by using different approaches, however, there are still some gaps to be improved.

Finally, in Ouriques et al. (2019), a systematic literature review was conducted. A narrative synthesis was applied in order to analyze the data. The authors illustrate the KM strategies in the hierarchical layers of software development companies that adopt ASD, through a conceptual classification framework, and

**Table 3**
Initial set of selected studies.

| References | Year | Nº primary studies | Study title |
|---|---|---|---|
| Neves et al. (2011) | 2011 | 14 | Knowledge creation and sharing in software development teams using Agile methodologies: key insights affecting their adoption |
| Cabral et al. (2014) | 2014 | 25 | Knowledge Management in Agile Software Projects: A Systematic Review |
| Paredes et al. (2014) | 2014 | 50 | Information Visualization for Agile Software Development Teams |
| Andriyani et al. (2017) | 2017 | 48 | Understanding Knowledge Management in Agile Software Development Practice |
| Borrego et al. (2017) | 2017 | 42 | Review of approaches to manage architectural knowledge in Agile Global Software Development |
| Indumini and Vasanthapriyan (2018) | 2018 | 12 | Knowledge Management in Agile Software Development — A Literature Review |
| Ouriques et al. (2019) | 2019 | 32 | Knowledge Management Strategies and Processes in Agile Software Development: A Systematic Literature Review |
| Al Hafidz and Sensuse (2019) | 2019 | 15 | A systematic literature review of improved knowledge management in agile software development |

**Table 4**
Concept of the studies according to the KM and agile perspectives.

| Perspectives | Concepts | Studies |
|---|---|---|
| KM | Sharing | Andriyani et al. (2017), Cabral et al. (2014), Neves et al. (2011), Paredes et al. (2014), Borrego et al. (2017), Ouriques et al. (2019), Al Hafidz and Sensuse (2019) and Indumini and Vasanthapriyan (2018) |
| | Capture/Creation | Neves et al. (2011) and Al Hafidz and Sensuse (2019) |
| | Knowledge types | Andriyani et al. (2017), Cabral et al. (2014), Ouriques et al. (2019), and Al Hafidz and Sensuse (2019) |
| | Reuse | Cabral et al. (2014) and Indumini and Vasanthapriyan (2018) |
| | Cycle (all activities) | Andriyani et al. (2017), Cabral et al. (2014), Paredes et al. (2014), Ouriques et al. (2019) and Al Hafidz and Sensuse (2019) |
| | Management strategies | Andriyani et al. (2017) and Ouriques et al. (2019) |
| | Architectural KM | Borrego et al. (2017) |
| | Knowledge representation | Indumini and Vasanthapriyan (2018) |
| Agile | Software artifacts | Andriyani et al. (2017), Cabral et al. (2014), Paredes et al. (2014), Borrego et al. (2017), and Al Hafidz and Sensuse (2019) |
| | Communication team | Andriyani et al. (2017), Cabral et al. (2014), Neves et al. (2011), Borrego et al. (2017), Ouriques et al. (2019) and Al Hafidz and Sensuse (2019) |
| | Collaborative process | Cabral et al. (2014), Ouriques et al. (2019) and Al Hafidz and Sensuse (2019) |
| | Agile practices | Andriyani et al. (2017), Neves et al. (2011), Borrego et al. (2017) and Indumini and Vasanthapriyan (2018) |
| | Architectural issues | Borrego et al. (2017) and Indumini and Vasanthapriyan (2018) |
| | Technological solutions/tools | Cabral et al. (2014), Borrego et al. (2017), Al Hafidz and Sensuse (2019) and Indumini and Vasanthapriyan (2018) |

provide comprehension concerning how the companies implement KM strategies with practices that promote the knowledge processes.

Once each study was deeply analyzed, we started to identify the key concepts addressed in each individual study, making it possible to determine how these concepts are related. This relationship is shown in the next stage of the synthesis method.

(4) ***Determining how the studies are related***

In the interest of a better understanding of the concepts found in the selected studies, we intend to answer the following research question in this stage: *What is the study concept presented according to KM and Agile perspectives?* In order to answer this question, we created Table 4 to present the perspectives and their concepts, together with the studies and concepts they have employed.

In relation to KM perspective, knowledge sharing is addressed by all studies. In Andriyani et al. (2017) and Paredes et al. (2014), for instance, one of the concepts is to promote the use of visualization techniques to improve sharing of knowledge. Agile principles promote producing less documentation. This fact leads the researchers to explore strategies of information representation in a leaner and efficiently, as is the case of Andriyani et al. (2017), Paredes et al. (2014). In KM, one of the main problems is how to represent knowledge so that it is easily shared (Souza, 2014). Visualization techniques minimize ambiguity and imprecision in interpreting shared information.

From the agile perspective, the two most investigated concepts were Software Artifacts and Communication Team. In Cabral et al. (2014), for example, one of the classifications is about problems and specific aspects of project documentation in agile methods. In agile projects, face-to-face communication is prioritized, instead of documentation. However, some primary studies returned in Cabral et al. (2014) suggest that even though the project is agile, if the team is distributed, there should be a greater focus on documents for knowledge sharing.

KM strategies were investigated by Andriyani et al. (2017) (included in our KM perspective). According to Andriyani et al. (2017), with respect to this concept, agile teams use: discussions (e.g. sharing requirements), artifacts (e.g. user stories), and visualizations (e.g. burn-down charts), to manage knowledge. The discussions, in particular, are verbal communications involving the

**Table 5**
Relation KM x agile values.

| KM activities | Agile values |
|---|---|
| Knowledge capture and/or creation | II, IV |
| Knowledge sharing and dissemination | I, III |
| Knowledge acquisition and application | II, IV |

interaction among agile team members willing to share knowledge. Therefore, in the agile perspective, we included Andriyani et al. (2017)'s work in "Communication Team".

Other concepts (purposes) and relationships that were identified are presented in Table 4. This stage helped us to determine how the studies are related. Thus, we started to work on mapping the studies into one another, which corresponds to stage five of the synthesizing method, presented below.

(5) *Translating the studies into one another*

Considering that the main goal of this stage is to compare metaphors among the studies and the fact that our research question relates agile values and agile practices, in this stage we present the relation created from stages 3 and 4. To construct the relations between KM and ASD, we first considered the definition of each KM process activity versus the Agile values definition, Scrum events and artifacts, and XP practices presented in the literature. As a result, we build Tables 5 and 6 (column 1 and 2) and Table 7 (column 1 and 2). This step was performed by two authors individually, and then several meetings with all authors were conducted to discuss each identified relationship. The discussion was based on arguments and supported facts and the disagreements were solved through consensus. Secondly, we created a list containing all metaphors (text fragments) from the selected secondary studies in the tertiary study that presented any relation between ASD and KM. Also, in this step, we separated these metaphors in (i) general metaphors: fragments that addressed a relationship between KM and ASD generally; (ii) Agile Values metaphors: fragments that addressed a relationship between KM and Agile values; (iii) Scrum metaphors: fragments that addressed a relationship between KM and Scrum specifically; and (iv) XP metaphors: fragments that addressed a relationship between KM and XP specifically. Finally, we checked our list of metaphors to see if any of them support our relationship findings constructed in the first step. Therefore, we validated Table 5 and also added specific and general citations to support the identified relationships in Tables 6 and 7 (see columns 4 and 5). This process was executed by one author and next, revised and discussed through consensus meetings with other authors i.e., any divergence between a relationship and citations were analyzed and discussed among the authors.

Following, we present the results of constructed relations between KM and Agile values, Scrum events and artifacts, and XP practices.

We focused on the three KM activities described in Section 2.1 and the agile values presented in Section 2.3. Table 5 illustrates the relation between KM activities and agile values. The agile values are referenced by their numbers as follows: (I) *Individuals and interactions over processes and tools;* (II) *Working software over comprehensive documentation;* (III) *Customer collaboration over contract negotiation;* and (IV) *Responding to change over following a plan.*

The study conducted by Neves et al. (2011) presents an analysis and evaluation of KM creation and sharing experiences in ASD teams. Through specific analysis, the study showed that the values (II) and (IV) are directly related to knowledge creation and the values (I) and (III) to knowledge sharing, as illustrated in Table 5.

The KM activity **"knowledge capture and/or creation"** is present in the agile values *Working software over comprehensive documentation* and *Responding to change over following a plan* due to the fact to the development of software requires the creation or capture of the knowledge to build it. The knowledge capture can be emerged from requirements elicitation Neves et al. (2011) or experiences from past projects. The knowledge creation also is present, for example, in every sprint execution when changes and adaptations are made to satisfy the client needs (Ouriques et al., 2019).

The most common form of sharing and disseminating knowledge is thought communication (Dalkir, 2005; North and Kumta, 2018). This fact leads us to understand that the value *Individuals and interactions over processes and tools* and *Customer collaboration over contract negotiation* are related to this KM activity of **"Knowledge sharing and dissemination"** (Neves et al., 2011; Andriyani et al., 2017; Ouriques et al., 2019). It is the exchange of knowledge through the agile team that will lead to solving impediments and the interaction with the client will lead to meet his/her business needs (Paredes et al., 2014; Andriyani et al., 2017).

In relation to **"Knowledge acquisition and application"** activities, the acquisition of knowledge is the comprehension, amplification, and articulation of knowledge in a way that it is internalized. Knowledge application refers to the real use of knowledge that has been captured or created (Dalkir, 2005). The agile value *Working software over comprehensive documentation* recognize the need for documentation, but it focuses more on working software; and to keep this focus, the agile team needs to articulate the existing knowledge and apply it in practice instead of relying only in documentation which could lead delays in development. The value *Responding to change over following a plan* refers to the application of the internalized knowledge of the agile team to be able to improve the product adding value to it. In other words, working the software development or responding to changes, for instance, experience is required. It is only possible to apply certain knowledge after it had been acquired and the best way to acquire knowledge is through experience (Cabral et al., 2014).

As mentioned earlier (Section 2.3), in order to demonstrate the relation between agile practices and KM activities, we opted to use Scrum and XP frameworks. Besides the eight initial sets of studies, we also adopted the Schwaber and Sutherland (2017) and Kent Beck's book Beck and Andres (2004) as the basis for this stage of our study.

### (5.1) *Translating the Scrum framework*

Table 6 presents the relation among each KM activity and Sprint Events and Scrum Artifacts as well as citations from the eight secondary studies included in our tertiary study that induce or confirm the respective relation presented. A more detailed version of Table 6 is available online on https://bit.ly/2nD4FNU (first tab/sheet of the spreadsheet). Four columns are shown in Table 6. The first one presents in each line one Scrum activity or artifact. The second one presents a brief description of the relationship between ASD and KM of each Scrum activity or artifact. This information was extracted from Schwaber and Sutherland (2017). The third and the fourth columns present specific citations (citations that directly support the relationship between one specific Scrum activity or artifact and KM activity) and general citations (citations that generally support the relationship between Scrum activities & artifacts and the KM activity) from our set of selected secondary studies.

Analyzing some relationships identified from Table 6, it is possible to say that the sharing and dissemination of activities are included in all Scrum artifacts and almost all Scrum events. Only in the Sprint Planning, they are not directly identified by

**Table 6**
Scrum events and artifacts x KM activities.

| SCRUM | Relation ASD and KM from SCRUM Guide | Specific citations | General citations |
|---|---|---|---|
| KM Activity: Knowledge capture and/or creation | | | |
| Sprint planning | The work to be performed in Sprint is planned by collaborative work of the entire team. New knowledge can be generated through discussion of the backlog items. | – | "...daily meetings and other activities of interaction among stakeholders are common. Knowledge is potentially stored in people's memory..." Cabral et al. (2014) "Physical forms that contain specific product features and project information, for example... Product backlog and Sprint backlog... ...code repositories..." Andriyani et al. (2017) |
| Daily scrum | Development team improves communication, eliminate other meetings, identify and remove development impediments, highlight and promote rapid decision making and improve the knowledge level. | "Face-to-face communication in daily meetings (SCRUM), …Neves et al. (2011). ...in a daily stand-up meeting agile teams clarify their cumulative work done by the team..." Andriyani et al. (2017) | |
| Development work | During the Development Work, development team members create and build increments. | – | |
| Product backlog | Product Backlog is an ordered list of everything that is known to be necessary for the product. It is the unique source of the requirements for any changes to be made to the product. | "... such as product backlog... ...were collectively classified as artifacts since they contained useful knowledge about the software requirements... The product backlog also helped capture product knowledge..." Neves et al. (2011) | |
| Sprint Backlog | Team modifies the Sprint Backlog (plan) during the Sprint. Backlog items are discussed and negotiated with the Product Owner. | – | |
| KM Activity: Knowledge sharing and dissemination | | | |
| Daily scrum | Team answer question and discuss what was done what will be done and any impediment to achieving the sprint goal. | "...daily meetings with the team (daily scrum) so that knowledge is shared inside the team" Cabral et al. (2014) | "...agile methodologies, such as SCRUM, improves organizational learning. ...increase tacit knowledge exchange...". "Verbal communication that involves interaction among agile team members to share knowledge..." Cabral et al. (2014) ...knowledge was shared during discussions... sprint review, testing, and small releases..." Andriyani et al. (2017) |
| Sprint review | Development team discusses what went well, what problems occurred and how these issues were resolved during Sprint. | – | |
| Sprint Retrospective | An evaluation is made on how the sprint was conducted in order to identify improvements to next Sprint. | "In a sprint retrospective project knowledge is shared..." Andriyani et al. (2017) "...retrospective meetings could support KS (Knowledge Sharing) Al Hafidz and Sensuse (2019) | |
| Development work | Development team implements functionality and technology. So, as Scrum promotes collaboration, during development work developers communicate whenever it is necessary. | "Agile teams to increase knowledge sharing when designing, developing..." Paredes et al. (2014) "Verbal communication that involves interaction among agile team members which aims to share knowledge..." Andriyani et al. (2017) | |
| Product backlog | Product Owner discusses the Product Backlog as it stands, and it is evaluated at each increment delivery. | "Artifacts in agile practices were commonly used to share product knowledge... (e.g. in the form product backlog)" Andriyani et al. (2017) | |
| Sprint Backlog | The Sprint Backlog is disseminated and discussed during the sprint and at the end of the sprint, it becomes an increment. | – | |
| Increment | The increment is shared (presented), reviewed and assessed in the Sprint Review. | – | |
| KM Activity: Knowledge acquisition and application | | | |
| Development work | New knowledge is acquired, and others are reinforced. The development teams have all necessary skills as a team to create the product increment. | – | – |
| Increment | An Increment is a body of inspectable, finished work that supports empiricism at the end of the sprint. The increment is a step towards a vision or a goal. | "Agile methods provide deliverables after each iteration, …, thereby facilitating interaction, trust and understanding between on-site customers and the developers" Neves et al. (2011) | |

considered literature. However, during this Scrum event there is indirect information sharing and dissemination between the Product Owner and the scrum team when they are discussing the Sprint purpose and the Product Backlog items that if completed, they will reach the Sprint goal (Schwaber and Sutherland, 2017).

Few Scrum artifacts and events are related to acquisition and application activities within the context of KM activity. We have identified only one event (development work) and one artifact (increment) that are directly related to this KM activity. This fact induces that more research on the acquisition and application of KM activity and ASD should be explored. Regarding increment, it is important to make some considerations: The increment is the accumulation of the value of all increments from previous sprints plus the sum of all product backlog items done (according to the established definition of "done") during the current sprint (Schwaber and Sutherland, 2017). The delivered increment from each sprint facilitates the understanding between the on-site customers and development Neves et al. (2011), promoting knowledge externalization between these parts since tacit knowledge from Scrum team can be materialized into a product functionality for the customers.

The event related to development is not explored in details in the Scrum Guide (Schwaber and Sutherland, 2017). It is mentioned, but no details are reported on the aspects of its practice. During the research and reflection about the relation between the KM activities and Scrum events, we could notice that the development work is present in all three activities of KM (see Table 6). This fact is justified by Scrum values that collaboratively construct software. The product backlog items "done" produced during the sprint and delivered at the end of the Sprint as an increment, are built mainly through source code implemented during the development work. The code generated during the development is not always transformed into explicit knowledge since the information from code is not easy to understand as well as design decisions are not clear by examining only the source code (Pfleeger, 2001; Galster and Babar, 2014). Nonaka and Takeuchi (1995) divides explicit knowledge into two forms: (i) documented–externalized knowledge documented in an informal way that over time could be vaporized; and (ii) formalized — knowledge documented following a standardized structure that can be easily retrieved when necessary. The scrum artifacts and code are usually documented explicit knowledge since usually they are documented without following standards or even these standards are not clear for who is involved in the development team or future stakeholders (Nonaka and Takeuchi, 1995; Borrego et al., 2019).

Concerning the tacit knowledge, according to Nonaka and Takeuchi (1995), it needs to be converted to explicit knowledge

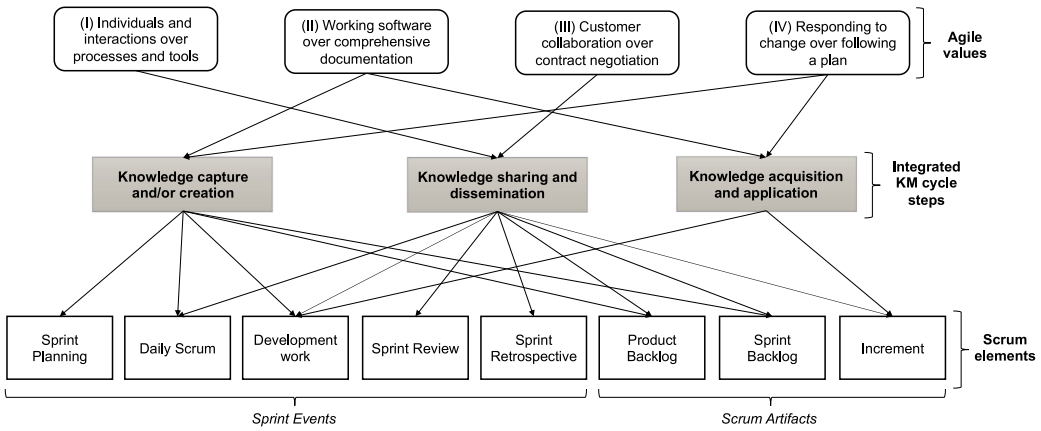


**Fig. 5.** Synthesis of Translations: Relation Map of the relationships between KM, agile values and scrum elements (events and artifacts).

(externalization) in order to avoid knowledge vaporizing. Some practices can be performed during the Scrum events to externalize knowledge Kamthan (2013), Borrego et al. (2017): *sprint planning* — document details of the discussion about each item of the sprint backlog in a platform where all teams have access; *daily scrum* and *development work* — after the daily scrum the participants can document the impediments and solutions discussed in the meeting as well as the developed solution in an internal Wiki, blog, or intranet; *sprint review* and *retrospective* — document the results, lessons learned, negative points, positives points and action plans that should be considered in the next Sprint or even in future similar projects. Although different practices for externalizing knowledge can be used, as previously presented, it is still common in ASD environment the staff be reluctant to document (Clear, 2003). As reported by Behutiye et al. (2017), ASD relies on team tacit knowledge and leans towards reducing the focus on requirements specification and documentation. One of the main characteristics of ASD is the short iterations and the quick delivery of working software. In ASD, developers face time pressure, mainly focus on the delivery of functionalities, and often do not consider the documentation, which can result in an increase in maintenance cost and effort (Mendes et al., 2016).

### (5.2) *Translating the XP Method*

Table 7 presents the relation between each KM activity and XP practices. This is similar to what we did for Sprint Events and Artifacts of Scrum; we included citations from the eight secondary studies included in our tertiary study that induce or confirm the respective relation presented. A more detailed version of Table 7 is available online on https://bit.ly/2nD4FNU (second tab/sheet of the spreadsheet).

There are four columns in Table 7. The first one presents in each line one XP practice. The second one presents a brief description of the relationship between XP practices and KM activities extracted from Kent Beck's book (Beck and Andres, 2004). The third and the fourth columns present specific citations (citations that directly support the relationship between one specific XP practice and KM activity) and general citations (citations that generally support the relationship between XP practices and one KM activity) from our set of selected secondary studies.

Regarding the relationships identified and presented in Table 7, it is possible to state that the sharing, dissemination, and assessment of activities are present in most practices of XP (6 out 10 practices considered in Table 7). One XP practice that has the strongest relationship with the activities of sharing and dissemination is pair programming as it is considered the main practice and the most cited by the researchers who study KM and XP (practice with more specific citations found — See Table 7 column "Specific Citations").

XP practices "Weekly Cycle" and "Quarterly Cycle" are directly related to the activities of sharing and dissemination as well as capturing and/or creation. This behavior was observed only in these two practices as both practices address development cycles, including plan meetings, project development discussions, and other interactions among stakeholders (developers, testers, clients, etc.) that allow creating and/or capturing the knowledge besides sharing it.

Likewise, with respect to knowledge capture and/or creation activities, there are only three practices related to acquisition and application activities. This fact reinforces that both these KM activities are not well explored yet in the XP context.

### (6) *Synthesizing translations*

According to meta-ethnography method Noblit and Hare (1988), in stage six, a translation synthesis is constructed. Thus, we created two relation maps that summarize the synthesis identified for Scrum and XP in relation to KM. The relation maps are presented in Figs. 5 and 6.

In Figs. 5 and 6 we evidenced all the concepts and their relationships identified in this research. Fig. 5 presents the synthesis of translations in relation to agile values, events and scrum artifacts, and KM activities. Fig. 6 presents the synthesis of translations between XP practices and KM activities.

Once this synthesis of translations has been created, we decided to collect opinions and/or impressions about it by conducting interviews with domain experts. To collect experts' opinions is not part of meta-ethnography method. However, we consider this process important to ensure the validity of the concepts and relations between ASD and KM identified, since the synthesis of translations was created from a specific literature, but we also deem it important to elicit impressions from a practical view of professionals who have been working for many years with agile development.

We conducted semi-structured interviews with four different ASD professionals. Semi-structured interviews allow for improvisation and exploration of the studied objects (Wohlin et al., 2012). The interviews were conducted with four ASD professionals at two different times. Scrum interviews occurred in June 2018 with three ASD professionals. In May 2019, XP interviews were conducted with three professionals, two of whom were the same that participated in the Scrum interview. Interviewees were chosen based on the deep experience with ASD and on their availability and willingness to participate in the research. Thus, for the XP practices interviews we changed one interviewee due to his unavailability to participate in the research. The interviewees have

**Table 7**
XP practices x KM activities.

| XP | Relation between XP practices and KM from Kent Beck's book | Specific citations | General citations |
|---|---|---|---|
| KM Activity: Knowledge capture and/or creation | | | |
| Stories | Stories serve to create the ideas, to place them in a context from which they can be more easily understood by listeners. | – | "They use the knowledge creation model proposed by Tuomi to analyze XP practices and state that these practices enhance knowledge creation..." Cabral et al. (2014) |
| Weekly cycle | To plan work for a week at a time in order to review progress to date, pick a week's worth of stories to implement this week, and break the stories into tasks. | "...customer involvement throughout the development cycles allow team members to discuss issues and create knowledge through continuous feedback." Ouriques et al. (2019) | |
| Quarterly cycle | Quarterly cycle verifies developed activities within a weekly cycle looking into the perspective of the entire project. Usually, the plan is discussed for a particular quarter, thus, enabling the customers to evaluate the project progress besides considering other activities with other stakeholders. | "...customer involvement throughout the development cycles allows team members to discuss issues and create knowledge through continuous feedback." Ouriques et al. (2019) | |
| KM Activity: Knowledge sharing and dissemination | | | |
| Sit Together | Sit together consists of the practice of having the team members close to each other encouraging conversation. | "...work together for long periods in a distributed context, facilitates knowledge transfer..." Ouriques et al. (2019) "...physical workspace affects knowledge transfer effectiveness." Ouriques et al. (2019) | |
| Whole team | Intense interactions among the team members are necessary for the health of the project. People need a sense of "team": we belong, we are in this together, we support each other's work, growth, and learning. What constitutes a "whole team" is dynamic. | "...the members to understand that they need to trust each other and share knowledge... This practice is pointed out... as valuable for knowledge sharing Ouriques et al. (2019)". "...groupware tools to enable frequent communication and build social ties. ...the team's social potential to share knowledge." Ouriques et al. (2019) | "...the interaction during the meetings promotes discussions... This practice allows team members to share knowledge..." Ouriques et al. (2019) "Walls that separate offices act as barriers for knowledge transfer. ...workspaces that integrate people and furniture that facilitate working together... provide closeness between members." Ouriques et al. (2019) "Process knowledge was shared during discussions..." Andriyani et al. (2017) |
| Pair programming | Pair programming deals with software development and two people who program simultaneously. The developers sit side-by-side and dialog between them. | "The pair programming technique as a form of sharing tacit knowledge..." Cabral et al. (2014) "...knowledge exchange effectiveness in pair programming." Cabral et al. (2014) "...most primary studies pay more attention to practices such as pair programming... ...these practices emphasize tacit knowledge sharing..." Andriyani et al. (2017) "...team members to share their knowledge." Ouriques et al. (2019) | |
| Weekly cycle | To plan work for a week at a time in order to review progress to date, pick a week's worth of stories to implement this week, and break the stories into tasks. | "...intensive communication facilitates knowledge transfer..." Ouriques et al. (2019) | |
| Quarterly cycle | Quarterly cycle verifies developed activities within a weekly cycle looking into the perspective of the entire project. Usually, the plan is discussed for a particular quarter, thus, enabling the customers to evaluate the project progress besides considering other activities with other stakeholders. | – | |
| Test-First programming | Test-First programming consists in writing a failing automated test before changing any code. It contributes to: delimitation of the scope of the code, focusing objectively and explicitly on what the functionality should do; coupling and cohesion since loosely coupled, highly cohesive code is easy to test; confidence in the quality of the code; development rhythm (test, code and refactor). | – | |
| KM Activity: Knowledge acquisition and application | | | |
| Informative workspace | Consists in make one's workspace about one's work. The workspace needs to be geared to the project. | – | – |
| Continuous integration | Integration all programmed code and test. | – | |
| Incremental design | Suggests that the most effective time to design is in the light of experience. In other words, the XP team should make the design of the system an excellent fit for the needs of the system that day. | "Agile methods provide deliverables after each iteration, …, facilitating interaction, trust and understanding between on-site customers and the developers" Neves et al. (2011) | |

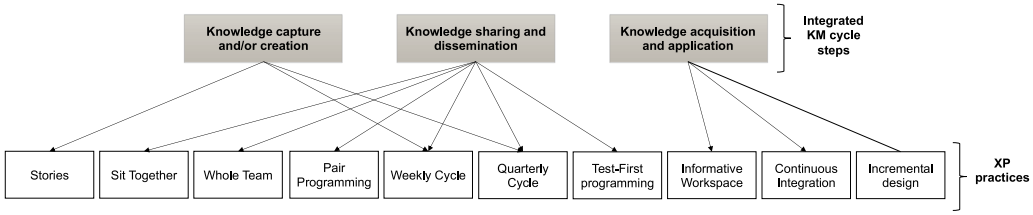


**Fig. 6.** Synthesis of Translations: Relation Map of the relationships between KM and XP practices.

on average 12 years of experience with ASD, focusing on Scrum and XP frameworks.

The interview questions were created considering the relationships in our synthesis of translations presented in Figs. 5 and 6. These questions were created in the affirmation format, for example, for the relationship between the concepts "**Knowledge capture and/or creation**" and "**Sprint Planning**" (see Fig. 5) the following statement was created: "*In agile methods, the Sprint Planning event is the time when the team meets to plan and organize the work to be done. It is at that moment that the internal knowledge is identified and created, that is, at this moment new knowledge can arise. In this activity, the team's tacit knowledge is captured.*" In order to answer each affirmation, the interviewee chose a degree of agreement considering a scale of 1 to 5 based on the Likert Scale method (Likert, 1932).

For each affirmation question presented earlier, we have created an open question to identify examples or reports of the respondent's life experience in relation to their response. At this moment, new questions spontaneously have arisen, which allowed for a better conversation flow and better information.

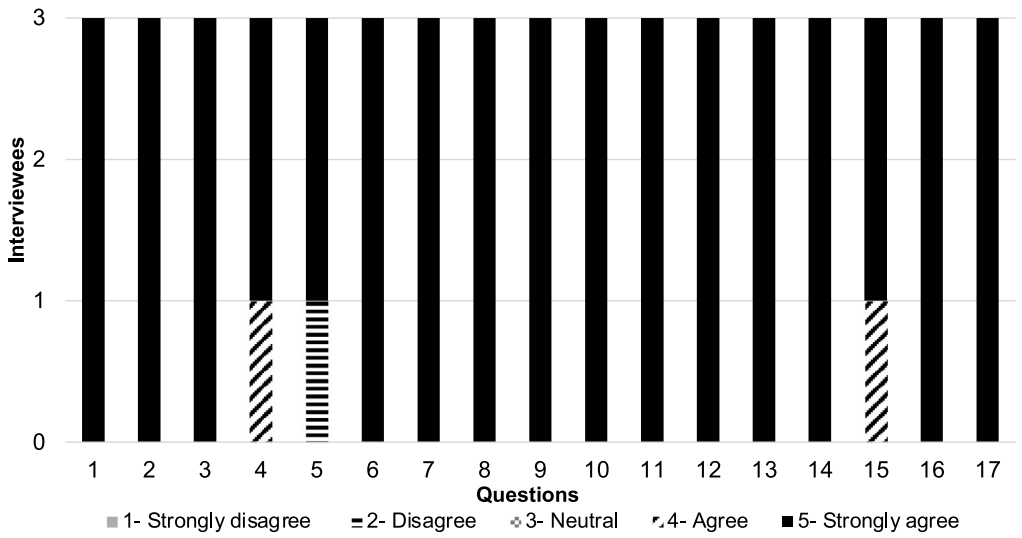


**Fig. 7.** Interview results for each question for the agile values, events and scrum artifacts, and KM activities based on Likert Scale method.

In relation to the statement previously presented (capturing tacit knowledge[2]), for example, the interviewees mentioned that brainstorming was the technique they most use to capture tacit knowledge. Brainstorming is a practice by which efforts are made to find a solution for a specific problem by gathering a list of ideas contributed by the team (Paetsch et al., 2003). Besides brainstorming, one of the interviewees reported performing quick meetings to discuss problems and during these meetings, they used sticky notes to capture and organize the tacit knowledge generated.

The first step during the interviews was to provide to the interviewee an explanation about each KM activity from Dalkir's KM cycle Dalkir (2005) to avoid bias or misunderstanding regarding the relationships. The interviews were conducted in a synchronous way (face to face) through video conference (via Skype or Hangouts). It is important to highlight that we created a questionnaire only to assist the interviewee's understanding and to assist the interviewer to conduct the interview. In this questionnaire, we include a mixture of open-ended and specific questions, designed to elicit information foreseen and also unexpected types of information. During the interview, the interviewees had the opportunity of choosing the level of agreement with the stated relationship, and also, they were asked to justify their answers. If they agree or not with one relationship, they were asked about their thoughts on the connection of the presented relationship with the questioned Dalkir's KM cycle activity or even another activity.

The interviews were conducted in the Portuguese language since it is the interviewees' native speaking language and lasted approximately one hour each. Their answers were transcribed for analysis and interpretation. All the questions with respect to the interviews can be accessed at https://bit.ly/2LVWzdl. Figs. 7 and 8 presents details on each interviewee answer for each question based on his/her level of agreement (Likert scale method).

With respect to the relationships impressions for the agile values, events and scrum artifacts, and KM activities created in Fig. 5, 14 relationships were confirmed from the three interviewees answers with scales equaling to 5; this represents 82.3% of the synthesis of translations relationships for the agile values, events, and scrum artifacts. Out of the other three questions in which there was no consensus on a full agreement, two interviewees pointed out conformance on a scale of 4 (11.8%), which is a strong indication of the relationships created. However, in one of the questions, the interviewee answered a scale of 2 for agreement (5.9%) (see Fig. 7 — Questions 4, 5, and 15). Basically, the question stated that in Sprint Backlog new explicit knowledge is generated and organized ("**Knowledge capture and/or creation**" x "**Sprint Backlog**"). We asked the interviewee why he/she did not fully agree with the statement, and the answer was that in his/her opinion most of the knowledge is generated in the sprint planning process. In sprint planning the tasks are discussed and explained, but in the sprint backlog, although there may be increases in knowledge, it is little relevant. The results already explained regarding the relationship impressions for the agile values, events, and scrum artifacts, and KM activities are summarized in Fig. 9 and presented individually in Fig. 7. The agreement of two area experts with the "**Knowledge capture and/or creation**" x "**Sprint Backlog**" relationship still indicates the relationship existence. Even so, this relationship was discussed in the threats to validity section (Section 5.4).

Regarding the relationships impressions for the XP practices and KM activities created in Fig. 6, 10 out of 12 relationships are confirmed by the following response pattern: two interviewees' answers with scales equaling to 5 and one interviewee answers with scales equaling to 4. This pattern represents 83.3% of the synthesis of translation relationships for the XP and it means that the majority of the XP practices are strongly connected with KM activities.

In the question stated that the practice "Sit Together" allows sharing and exchange of knowledge among the team members ("**Knowledge sharing and dissemination**" x "**Sit Together**"), two interviewees pointed out conformance on a scale of 4 and one on a scale of 5 (see Fig. 8 — Question 2). Both who pointed out 4 mentioned in the comments that the practice of sitting together is not always productive, the practice needs to be implemented carefully in order to keep all the involved focused and productive.

All interviewees have already experienced the practice of pair programming by themselves and with their teams. Regarding the question about the relation between "**Knowledge sharing and dissemination**" and "**Pair programming**" two interviewees fully agreed with this relationship and mentioned in the comments that during the pair programming "there is an intense exchange of knowledge through effective communication and collaborative

[2] Tacit knowledge can be captured, for example, using brainstorming sessions or road maps. Through these techniques, team members quickly meet to solve certain day-to-day problems. In these meetings, lessons learned or experiences of each individual can be passed on to the group.

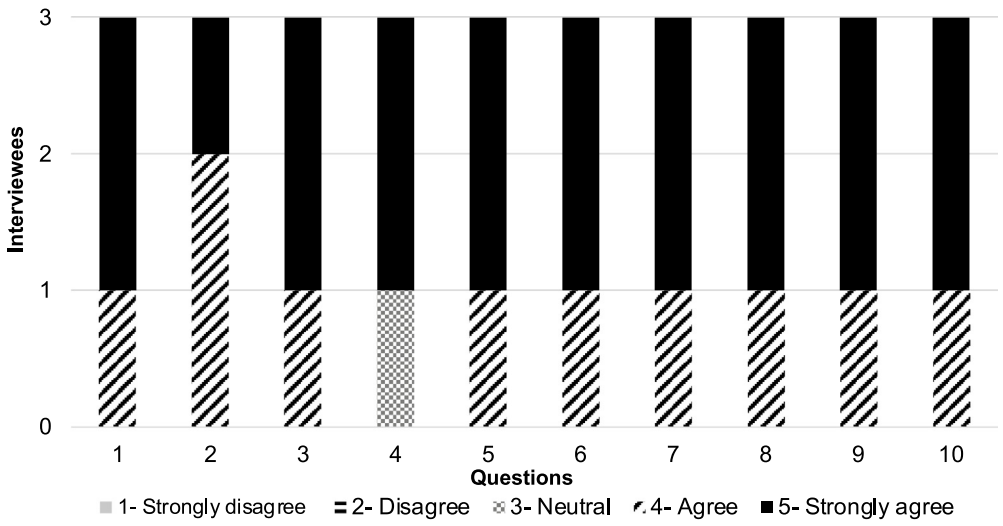


**Fig. 8.** Interview results for each question for the XP practices and KM activities based on Likert Scale method.

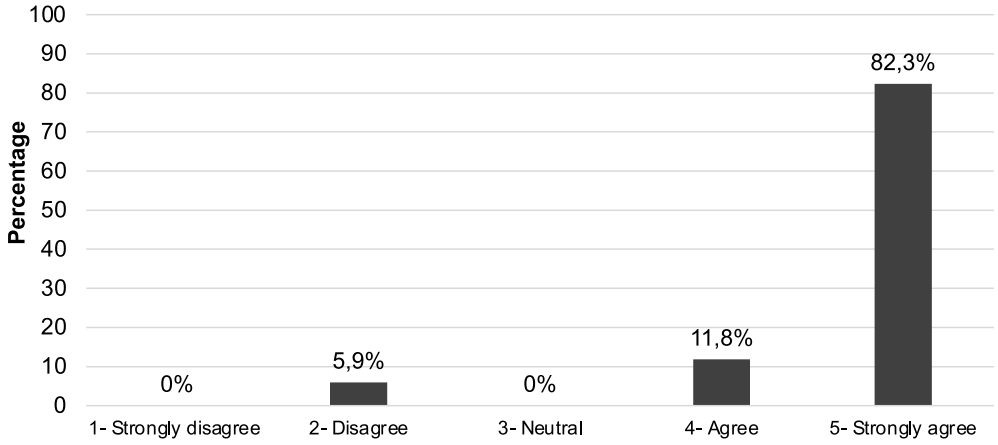


**Fig. 9.** Summarizing the interview results for the relationships impressions for the agile values, events and scrum artifacts, and KM activities based on Likert Scale method.

work" and that this practice "collaborate with team knowledge". However, one interviewee pointed out conformance on scale 3 (see Fig. 8 — Question 4) mentioning in the comments a relevant remark: "I believe that practicing pair programming 100% of the time loses productivity. The team needs to be trained and integrated to avoid that knowledge stays only with part of the team, consequently, the knowledge is not shared and disseminated properly".

The interviewees also made important positive notes and reports of the experiences they went through. Regarding some Scrum events and artifacts some interesting points besides the already mentioned ones follow:

- **Daily meeting:** "The experiences of the project participants are a very important factor for the success of a daily meeting. During the exchange of experience between the participants, a lot is added to the team because discussions and reflections are promoted in this activity. However, in some cases, some members prefer not to share information or postpone sharing, which could be a problem".
- **Development work:** "A lot of knowledge is generated during development, but due to the accelerated routine this knowledge ends up not being made explicit (being documented in the project wiki, for example)".
- **Retrospective:** "The purpose of the retrospective is to share knowledge about the team's strengths and weaknesses and to generate improvement actions, however, due to the lack of documentation and monitoring of the application of the knowledge generated in this meeting, the improvement ends up happening rarely due to the application of improvements in practice is a laborious and time consuming process".
- **Increment:** "The increment is the result of the acquisition and application of knowledge during the Sprint. Also, new knowledge is acquired and applied to each increment".

With respect to the XP practices, some interesting points were highlighted by the interviewees about the following practices.

- **Stories:** "There is the knowledge that the team already has, so more experienced teams have greater ease and agility to collaborate in detailing the requirement. There is also technical knowledge that the client does not have and so the team, when multidisciplinary, ends up adding technical knowledge to the requirement".
- **Weekly cycle and quarterly cycle:** "In practice what should drive the cycle size is the pattern: Build, Measure and Learn. In other words, the weekly and quarterly cycles depend on the team size, project scope, and when there are dependencies between teams and suppliers (e.g., points of integration with third-party solutions). Each project is different and must be evaluated according to a plan since not always what

is generated within the week is functional and thus some weeks have no deliveries for the external customer, only for an internal customer".

- **Test-first programming and continuous integration:** "In addition to test-first programming practice, continuous integration practices (such as DevOps) make a huge difference in any development project because it decreases the deployment time and consequently the cost, also helping in improving the project quality. They must be automated and constant. However, in the beginning, training and developers' commitments are necessary".
- **Incremental design:** "The development must be guided by the knowledge acquired during the development without BDUF (Big Design Up Front) and using techniques of emergent architecture".

In summary, all interviewees emphasized in the open questions, that the creation or transfer of knowledge within the companies that work with ASD are of great importance. For them, the software is knowledge transformed into important decisions in the organizations, for example, design decisions or code. Therefore, it is fundamental to take into account this information and keep it inside the company. Regarding the source code, although one of the interviewees mentioned that knowledge can be transformed into code, as previously discussed (see Section 5.1), the information from the source code is often not easy to understand to be retrieved later. According to Martínez-García et al. (2020), in software companies knowledge can be found in source code. However, in agile development, since the documentation is not a priority, knowledge is prone to disappear. If we consider teams that work remotely, for example, it can often happen that teammates are unable to answer project questions. In this situation, agile developers usually attempt to obtain answers by analyzing source code, which is time consuming, especially if the code is not self-documented (Borrego et al., 2019). For that reason, developing effective ways of managing software knowledge in agile environments, like code itself, has become of interest to area researchers. Currently, it is possible to find targeted research in the literature to reduce the vaporization in an agile environment (Borrego et al., 2016, 2019; Martínez-García et al., 2020).

One interviewee also emphasized that the creation and transfer of knowledge in an ASD organization help to reduce the high costs with employee changes. The experts mentioned that companies want to be agile, but they do not understand the effort in relation to the change that needs to happen. Companies are very much attached to contract, scope and deadline, but what the companies should do is to deliver the value (what the customer needs).

(7) ***Expressing the synthesis***

In this last stage, the objective is to report the synthesis results. According to Silva et al. (2013), the target audience is the research community interested in performing synthesis of empirical research using meta-ethnography. Thus, we believe that our synthesis can help to direct researchers in future research providing a direction to appropriately position new research activities.

## 5. Discussion

In this section, we discuss the findings of our study regarding the methodological approach adopted, the obtained results, and the study contributions.

### 5.1. Methodological approach

We began this study intending to answer the following question "*How do agile values and practices relate to KM activities?*". The meta-ethnography method was used to identify the existing synthetics between these areas. As pointed out by Silva et al. (2013), the meta-ethnography requires several readings of the initial set of studies, data extraction, consensus meeting, and verifying potential inconsistencies in interpretations. These tasks are time consuming and require maturity in the research team. However, although the method is not simple to apply, we believe that we have been able to answer the research question employing a synthesis of translations between KM, agile values, Scrum elements (events and artifacts), and XP practices considered plausible for comprehension. This synthesis of translations created can contribute mainly to researchers interested in understanding the relationships in this domain: KM and ASD.

A synthesis study focuses primarily on primary studies. However, we realized that the method could also be applied in secondary studies returned from a tertiary study. Even working with eight studies in the initial set, when necessary we consulted some of 212 primary studies returned in our eight secondary studies.

### 5.2. Obtained results

In relation to the study results, we can say that sharing is a KM activity that is strongly related to ASD and very much investigated in Scrum and XP frameworks. This makes sense since ASD prioritizes the exchange of information and communication among teams. Agile values promote a focus on the people involved in a project and how they interact and communicate. The communication is strongly related to KM, and in agile teams, the focus is tacit knowledge. Also, noteworthy that creating knowledge is a strongly present activity in Scrum, but not to the same degree in XP.

In this research, we note that the activity of acquisition and application has not been investigated like the other activities from both Scrum and XP. Knowledge applying activity is the last step of the KM cycle which suggests that the captured and shared knowledge must ultimately be used in the organization.

In the diagnostic that we conducted in 13 software development companies, the activity of using knowledge (Acquisition and Application) had the better performance, that is, the use of knowledge had greater attention within organizations through the practical view of professionals in the area. In this third stage of the KM integrated cycle (see Fig. 1), the knowledge is put to actual use. According to Dalkir (2005), knowledge eventually is made accessible to all in the organization, with an implicit assumption that they will be used. The explicit knowledge can be disseminated among the organization and finally be internalized, completing the last dimension of the SECI model (Socialization, Externalization, Combination, and Internalization) (Nonaka and Takeuchi, 1995). The use of explicit knowledge in these organizations can be observed in several activities such as during the code development, architecture definition and design, client requirements, among others, as these activities have their knowledge documented (externalized). Despite the diagnostic conduction was preliminary, the results made it possible to understand how KM activities are present in agile and traditional software development from a practical view and that the results may be different from those presented in the literature, as can be seen in this study.

By conducting the meta-ethnography process, enabling the analysis of the literature, it was possible to observe that the biggest challenge faced by software organizations is concentrated on how to capture and share knowledge. On the other hand,

this same view was not observed during the conduction of the diagnostic; in practice, the use of knowledge is the main point of attention for the investigated organizations.

We can infer that software organizations are concerned with how to use the knowledge generated. However, research on KM and ASD is more focused on how to transfer knowledge. This result can be important to direct new research in the area considering the interest of software companies.

### 5.3. Contributions

The major contribution of this study was to provide an understanding of how specific dimensions of KM in ASD can help area practitioners effectively manage the knowledge in everyday agile actions. Several organizations that adopt agile methodologies face a problem related to management and organizational knowledge retention. The agile premise is guided by contributory and collaborative work as well as knowledge evolution. The clarification of how KM is present in each agile value, Scrum event, and artifacts allows a reflection on how much knowledge has been created, shared, and applied during ASD. This reflection enables organizations to explore more each KM cycle phase, consequently, contributing to delivery with greater value to the client.

As highlighted in the introduction, agile practices and KM present common activities. Although the domains are different, they have many similarities. A professional in the field could infer the relationships between these two domains from their experience. However, there has been no research that formally presents the intersection of these two areas. Conducting meta-ethnography allowed to analyze KM and Agile with a high level of rigor and thus defend a given hypothesis.

In addition, the conduction of the meta-ethnography method can also be considered a contribution. According to Fu et al. (2019), there is a need of investing in gaining knowledge and experience in applying meta-ethnography. The high quality meta-ethnography can make significant contributions to empirical software engineering and we believe that our study presents a new example of how to relate two different areas as well as how to apply the method in practice. Therefore, this study can also support the improvement in the use of meta-ethnography in software engineering as well as the advance of future empirical research in this context.

Based on our experience using meta-ethnography in this study, we report to follow the main **advantages** and **challenges** identified. Among the main **advantages** we highlight: (a) as already mentioned in the previous paragraph, we believe that this study presents a practical example of how to synthesize two different areas as well as how to apply the meta-ethnography method for this purpose; (b) we presented that it is possible to make adjustments/adaptations in the method. Usually, the method is used to analyze primary studies, and in this study, in particular, our set of studies set assembled by secondary studies. However, when necessary, we looked for primary studies; (c) although KM and ASD have very similar characteristics, the study we conducted presents with rigor and formalism how these areas are related; and (d) the different team views (researchers and industry professionals) who conducted this study made possible to discuss the existing relationships in the areas studied, making the conclusions more realistic.

In relation to the main **challenges**, we can mention: (a) as highlighted by Fu et al. (2019), there is a lack of methodological reference materials and guidelines for performing meta-ethnography; (b) it is a laborious method and requires the execution of several long steps; (c) the method involves a team to discuss how two areas are related and this requires time, dedication and deadline available for meetings with everyone involved, which is not always easy to achieve; and (d) secondary studies did not always make information clear for already presenting a summarization. In some moments it was necessary to look for primary studies. However, the large number of primary studies has made it more time-consuming to adopt meta-ethnography.

### 5.4. Threats to validity

In this Section, we report the main threats to the validity of our study as well as their mitigation alternatives.

**Internal validity.** One threat to validity in this study could be the research method used. Since meta-ethnography is an interpretive approach to synthesis, we addressed the validity and reliability of our synthesis from an interview with four ASD experts from a practical aspect.

**Construct validity.** Some relationships between KM, agile values, and practices may not have appeared in our relation map. This is because we considered only eight secondary studies, Scrum guide, and Kent Beck's book. An alternative to this limitation would be to increase the set of references considered during the relationships construction and to consider a greater number of interviewees for the interview process.

**External validity.** A larger number of interviews could be consulted to improve this research quality; however, given the interviewees' profile, we believe these interviews can already bring important observations to the research. Even so, we intend to conduct a survey in order to continue validating our synthesis.

**Conclusion validity.** The results and the relationships between KM, agile values, and practices presented in our study could provide imprecise conclusions. In order to mitigate this threat, we based our conclusions on available references, and we performed interviews with domain experts. Also, all data from the interview were recorded and analyzed by at least two different authors and discussed among them in order to mitigate misinterpretation.

## 6. Conclusions

In this study, we reported on the results from an analysis on KM and ASD methodology using the meta-ethnography method. We applied the seven phases of meta-ethnography analysis method on eight studies selected from a tertiary study on KM and ASD. Then, the syntheses identified in these areas investigated were analyzed based on interviews with three ASD consultants.

Agile practices and KM present common activities that can encourage software organizations to promote KM activities. The exploration of KM activities during ASD can improve team learning and collaborate with the evolution of organizational knowledge leading to deliveries with greater value and consequently increase customer satisfaction. The most common activity between these two areas is knowledge sharing. In Cabral et al. (2014), they presented that there is still a gap about what emerges from the intersection of these two areas that need further clarification. In our investigation, we believe that the acquisition and application activities are research topics that can be explored more.

A richer investigation with better mechanisms to perceive KM in ASD forms part of future work. We intend to complement our research with other empirical methods, for example, we intend to conduct a survey in software organizations and to extend this analysis by adopting other agile methodologies. In addition, we intend to realize a deeper study addressing KM diagnostic in software companies to understand in detail how software organizations deal with KM in practice as well as perform statistical analysis of the data results reported by the companies in relation to KM practices.

## CRediT authorship contribution statement

**Bianca Minetto Napoleão:** Conceptualization, Supervision, Formal Analysis, Validation, Resources, Writing - original draft. **Érica Ferreira de Souza:** Conceptualization, Methodology, Supervision, Formal Analysis, Validation, Resources, Writing - original draft, Project administration, Funding acquisition. **Glauco Antonio Ruiz:** Formal Analysis, Validation, Resources, Writing - original draft. **Katia Romero Felizardo:** Resources, Writing - original draft, Visualization. **Giovani Volnei Meinerz:** Writing - review & editing. **Nandamudi Lankalapalli Vijaykumar:** Writing - review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

Dr. Érica Ferreira de Souza is supported by a research grant from the Brazilian funding agency CNPq, Brazil with reference number: 432247/2018-1.

## References

Abdullah, R., Eri, Z.D., Talib, A.M., 2011. A model of knowledge management system in managing knowledge of software testing environment. In: 5th Malaysian Conference in Software Engineering. MySEC'11. pp. 229–233.

Al Hafidz, M.U., Sensuse, D.I., 2019. A systematic literature review of improved knowledge management in Agile Software Development. In: 2nd International Conference on Software Engineering and Information Management. ICSIM 2019. New York, NY, USA, pp. 102–105. ISBN: 978-1-4503-6642-7.

Andrade, J., Ares, J., Martinez, M.-A., Pazos, J., Rodriguez, S., Romera, J., Suarez, S., 2013. An architectural model for software testing lesson learned systems. Inf. Softw. Technol. 55, 18–34.

Andriyani, Y., Hoda, R., Amor, R., 2017. Understanding knowledge management in agile software development practice. In: International Conference on Knowledge Science, Engineering and Management (KSEM). Melbourne, VIC, Australia, pp. 195–207.

Basili, V.R., Caldiera, G., Rombach, H.D., 1994. The experience factory. In: Marciniak, J. (Ed.), Encyclopedia of Software Engineering. John Wiley, New York, pp. 469–476.

Beck, K., Andres, C., 2004. Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, ISBN: 0321278658.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al., 2001. Manifesto for Agile Software Development. http://www.agilemanifesto.org/. (Accessed April, 2021).

Behutiye, W., Karhapää, P., Costal, D., Oivo, M., Franch, X., 2017. Non-functional requirements documentation in Agile Software Development: Challenges and solution proposal. In: Felderer, M., Méndez Fernández, D., Turhan, B., Kalinowski, M., Sarro, F., Winkler, D. (Eds.), Product-Focused Software Process Improvement. Springer International Publishing, Cham, pp. 515–522.

Bjørnson, F.O., Dingsøyr, T., 2008. Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used. Inf. Softw. Technol. 50, 1055–1068.

Borrego, G., Morán, A.L., Cinco, R.R.P., Rodríguez-Elias, O., García-Canseco, E., 2017. Review of approaches to manage architectural knowledge in Agile Global Software Development. IET Softw. 11 (3), 77–88.

Borrego, G., Morán, A.L., Palacio, R., Rodríguez, O.M., 2016. Understanding architectural knowledge sharing in AGSD teams: An empirical study. In: 11th International Conference on Global Software Engineering (ICGSE), pp. 109–118.

Borrego, G., Morán, A.L., Palacio, R., Vizcaíno, A., García, F.O., 2019. Towards a reduction in architectural knowledge vaporization during agile global software development. Inf. Softw. Technol. 112, 68–82.

Bukowitz, W., Williams, R.L., 2000. The Knowledge Management Fieldbook. Financial Times Prentice Hall, Great Britain.

Cabral, A., Ribeiro, M., Noll, R., 2014. Knowledge management in agile software projects: A systematic review. J. Inf. Knowl. Manag. 13, 1–37.

Clear, T., 2003. Documentation and agile methods: striking a balance. ACM SIGCSE Bull. 35, 12–13.

CollabNet VersionOne, 2019. 13th Annual State of Agile Report. Tech. Rep., https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508.

Cruzes, D.S., Dybå, T., 2011a. Recommended steps for thematic synthesis in software engineering. In: Empirical Software Engineering and Measurement (ESEM), pp. 275–284.

Cruzes, D.S., Dybå, T., 2011b. Research synthesis in software engineering: A tertiary study. Inf. Softw. Technol. 53 (5), 440–455.

Da Silva, F.Q.B., Cruz, S.S.J.O., Gouveia, T.B., Capretz, L.F., 2013. Using meta-ethnography to synthesize research: A worked example of the relations between personality and software team processes. In: Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on. IEEE, pp. 153–162.

Dalkir, K., 2005. Knowledge Management in Theory and Practice. Elsevier, USA.

Davenport, T.H., Prusak, L., 2000. Working Knowledge, second ed. Harward Business School Press, Boston, USA.

Esteves, S.R.M., 2017. Requisitos de software funcionais para o desenvolvimento de plataforma digital de diagóstigo da gestão do conhecimento nas organizações (Masters dissertation). UniCesumar, Maringá, Paraná, Brazil, (in portuguese).

Farrukh, M., Tariq, M.A., 2017. Agile methodology: Hybrid approach scrum and XP. Int. J. Sci. Eng. Res. 8 (4), 1405–1409.

Fonseca, A.F., 2006. Organizational Knowledge Assessment Methodology, second ed. World Bank Institute, Washington.

Fu, C., Zhang, H., Huang, X., Zhou, X., Li, Z., 2019. A review of meta-ethnographies in software engineering. In: Evaluation and Assessment in Software Engineering (EASE'19).

Galster, M., Babar, M.A., 2014. Empirical study of architectural knowledge management practices. In: 2014 IEEE/IFIP Conference on Software Architecture, pp. 239–242.

Hannes, K., Lockwood, C., 2011. Synthesizing Qualitative Research: Choosing the Right Approach. Springer.

Indumini, U., Vasanthapriyan, S., 2018. Knowledge management in Agile Software Development - A literature review. In: National Information Technology Conference (NITC), Colombo, Sri Lanka.

Kamthan, P., 2013. On the role of Wiki for managing knowledge in agile software development. In: 2013 International Conference on Collaboration Technologies and Systems (CTS), pp. 622–623.

Kitchenham, B., Charters, S., 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. Tech. Rep. EBSE 2007-001, Keele University and Durham University, UK.

Levy, M., Hazzan, O., 1999. The knowledge life cycle. In: ICM Conference on KM, Miami, FL.

Levy, M., Hazzan, O., 2009. Knowledge management in practice: The case of agile software development. In: ICSE Workshop on Cooperative and Human Aspects on Software Engineering. ICSE CHASE'09. Vancouver, BC, Canada, pp. 60–65.

Li, X., Zhang, W., 2012. Ontology-based testing platform for reusing. In: 6th International Conference on Internet Computing for Science and Engineering (ICSE'12), pp. 86–89.

Likert, R., 1932. A technique for the measurement of attitudes. Arch. Psychol.

Maciel, C.P.C., 2019. Knowledge Management Diagnostics in Software Development Companies (Masters dissertation). Federal University of Technology, Cornélio Procópio, Paraná, Brazil.

Martínez-García, J.R., Castillo-Barrera, F., Palacio, R.R., Borrego, G., Cuevas-Tello, J.C., 2020. Ontology for knowledge condensation to support expertise location in the code phase during software development process. IET Softw. 14 (3), 234–241.

Mendes, T., Farias, M.A., Mendonça, M., Soares, H.F., Kalinowski, M., Spínola, R., 2016. Impacts of agile requirements documentation debt on software projects: a retrospective study. In: SAC '16.

Meyer, M., Zack, M., 1996. The design and implementation of information products. Sage Publ. Inc 37 (3), 43–59.

Nejati, M., 2010. Knowledge management performance evaluation: Challenges and requirements for organizations. Tech. Technol. Educ. Manag. 5, 251–254.

Neves, F.T., Rosa, V.N., Correia, A.M.R., Neto, M.d.C., 2011. Knowledge creation and sharing in software development teams using Agile methodologies: Key insights affecting their adoption. In: 6th Iberian Conference on Information Systems and Technologies (CISTI), 1–6.

Noblit, G., Hare, R., 1988. Meta-Ethnography. Sage Publications Inc, Newbury Park, Calif.

Nonaka, I., Takeuchi, H., 1995. The Knowledge Creation Company: how Japanese Companies Create the Dynamics of Innovation. Oxford University Press, New York.
North, K., Kumta, G., 2018. Knowledge Management: Value Creation Through Organizational Learning, second ed. John Wiley & Sons.
O'Leary, D.E., 1998. Enterprise knowledge management. IEEE Comput. Mag. 54–61.
O'Leary, D., Studer, R., 2001. Knowledge management: an interdisciplinary approach. IEEE Intell. Syst. 16.
Ouriques, R.A.B., Wnuk, K., Gorschek, T., Svensson, R.B., 2019. Knowledge management strategies and processes in agile software development: A systematic literature review. In: International Journal of Software Engineering and Knowledge Engineering, pp. 345–380.
Paetsch, F., Eberlein, A., Maurer, F., 2003. Requirements engineering and agile software development. In: WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003, pp. 308–313.
Paredes, J., Anslow, C., Maurer, F., 2014. Information visualization for Agile Software Development teams. IEEE Working Conference on Software Visualization, pp. 157–166.
Pfleeger, S.L., 2001. Software Engineering: Theory and Practice, second ed. Prentice Hall PTR.
Pressman, R.S., 2014. Software Engineering: A Practitioner's Approach, eighth ed. McGraw-Hill.
Rodriguez-Elias, O.M., Martínez-García, A.I., Vizcaíno, A., Favela, J., Piattini, M., 2008. A framework to analyze information systems as knowledge flow facilitators. Inf. Softw. Technol. 50, 481–498.
Ruiz, G.A., Napoleão, B.M., Souza, E.F., Felizardo, K.R., Meinerz, G.V., Silva, P.R., Vijaykumar, N.L., 2018. Using meta-ethnography to synthesize research on knowledge management and agile software development methodology. In: 17th Brazilian Symposium on Software Quality (SBQS), Curitiba, Paraná, Brazil, pp. 230–239.
Santos, V., Salgado, J.G., Souza, E.F., Felizardo, K.R., Vijaykular, N.L., 2019. A tool for automation of knowledge management diagnostics in software development companies. In: Brazilian Conference on Software: Theory and Practice (CBSoft) - Tools Session, Salvador, Bahia, Brazil.
Schön, E.M., Escalona, M., Thomaschewski, J., 2015. Agile values and their implementation in practice. J. Interact. Multimedia Artif. Intell.
Schwaber, K., Sutherland, J., 2017. The scrum guide. https://www.scrumguides.org/download.html.
Silva, F., Cruz, S., Gouveia, T., Capretz, L., 2013. Using meta-ethnography to synthesize research: A worked example of the relations between personality and software team processes. In: 7th International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 153–162.
Souza, E.F., 2014. Knowledge Management Applied to Software Testing: An Ontology Based Framework (Ph.D. thesis). Instituto Nacional de Pesquisas Espaciais (INPE), Brazil.
Souza, E.F., Falbo, R.A., Vijaykumar, N.L., 2015. Knowledge management initiatives in software testing: A mapping study. Inf. Softw. Technol. 57, 378–391.
Sutherland, J., Schwaber, K., 2012. The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework. p. 224, 2011.
Terra, J.C.C., 2001. Gestão do Conhecimento: O Grande Desafio Empresarial, second ed. Negócio Editora, São Paulo.
Vasanthapriyan, S., Tian, J., Xiang, J., 2015. A survey on knowledge management in software engineering. In: International Conference on Software Quality, Reliability and Security Companion (QRS-C). Vancouver, BC, Canada, pp. 237–244.
Wendorff, P., Apshvalka, D., 1998. The knowledge management strategy of agile software development. In: 6th European Conference on Knowledge Management. Univ. of Limerick, Ireland, pp. 607–614.
Wiig, K., 1993. Knowledge management foundations: thinking about thinking - how people and organizations create, represent, and use knowledge. Schema Press, Arlington, TX.
Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A., 2012. Experimentation in Software Engineering. Springer Science & Business Media.

**Bianca Minetto Napoleão** holds a degree in Computer Engineering (2017) and an M.Sc. in Informatics (Applied Computing) (2019) from the Federal University of Technology — Paraná (UTFPR), Brazil. Her main research interests are systematic literature review and software development process including agile and open source practices.

**Érica Ferreira de Souza** holds a degree in Data Processing from Faculdade de Tecnologia do Estado de São Paulo (2005). She has Masters (2010) and Ph.D. (2014) degrees from the National Institute for Space Research (INPE) in Brazil. She is a Professor at Federal Uniersity of Technology — Paraná (UTFPR), Brazil. She has experience in software engineering and knowledge management. Her main research interests include software testing, knowledge management applied in software engineering and systematic literature review.

**Glauco Antonio Ruiz** holds a degree in Computer Science (2004) and a postgraduate degree in Information Technology Administration (2012) from the University of West Paulista (UNOESTE — Presidente Prudente/SP/Brazil), Mastering in Computer Science (Applied Computing) (2020) from the Federal University of Technology – Paraná (UTFPR), Brazil. He has worked for 15 years in software development areas and currently as Technology and Innovation Manager for 5 years. He has also been teaching at a university for 9 years. His main research interests are systematic literature review and knowledge management in agile software development.

**Katia Romero Felizardo** received the Post-Doctoral degree in Computer Science from the University of São Paulo (ICMC-USP), in 2014. Actually, she is Assistant Professor in Software Engineering at Federal University of Technology – Paraná (UTFPR), Brazil. Her research interests are in systematic literature review, information visualization and visual data mining.

**Giovani Volnei Meinerz** holds a Ph.D. in Eletronic and Computer Engineering from Aeronautics Institute of Technology (ITA), Brazil, in 2011, partially executed at Berlin University of Technology (TU-Berlin), Germany. Received his M.Sc. degree in Eletronic and Computer Engineering also from ITA, Brazil, in 2005 and, in 2001 he got his Bachelor on Computer Science from Regional Integrated University of High Uruguay and Missions (URI), Brazil. Currently he is adjunct professor at Federal University of Technology — Paraná (UTFPR), Brazil. His main research area of interests is Data Science, with focus on (i) big volume of NoSQL data management technologies, (ii) machine learning modeling techniques, and (iii) natural language processing tasks for extraction of sentiment analysis purposes.

**Nandamudi Lankalapalli Vijaykumar** holds an M.Sc. (1984) in Applied Computing from the Instituto Nacional de Pesquisas Espaciais (INPE) and a Ph.D. (1999) in Informatics from the Instituto Tecnológico de Aeronáutica (ITA). He has been an employee of INPE since 1978 at the Laboratório Associado de Computação e Matemática Aplicada (LAC) working on performance modeling and formal modeling for software testing. Currently, he is a Visiting Associate Professor at Federal University of São Paulo UNIFESP in São José dos Campos, Brazil.