Understanding and Supporting Knowledge Management in Agile Software Development

Raquel Ouriques
Understanding and Supporting Knowledge Management in Agile Software Development

Raquel Ouriques

Licentiate Dissertation in Software Engineering

Department of Software Engineering
Blekinge Institute of Technology
SWEDEN
The mind that opens to a new idea never comes back to its original size
- Albert Einstein
Abstract

**Background.** Agile Software Development (ASD) promises agility and flexibility in dealing with uncertainty by prioritizing interaction between people supported by informal communication and knowledge sharing. The lack of practices to manage the knowledge as a resource might jeopardize the application of knowledge in the production of goods and service. The utilization of Knowledge Management (KM) strategies can significantly support achieving and sustaining competitive advantage and brings several benefits to software development. However, how to manage knowledge in ASD is still not well understood or investigated.

**Objectives.** The main objective of this thesis is providing a different perspective on directions that KM can take to improve knowledge-based resource (KBR) management in ASD. The detailed objectives are: (i) Understand the current ASD environment regarding KM; (ii) Identify KBRs in ASD and its implications for KM; and (iii) Provide an initial set of variables to evaluate knowledge criticality of knowledge items in ASD.

**Method.** We use a mixed-methods approach to address the objective of this thesis. The methods selected to conduct the studies include systematic literature review, grounded theory, and improvement case study. The data collection comprised a literature review, semi-structured interviews, and practitioners’ feedback through static validation.

**Results.** From our SLR we observed that that KM strategies in ASD promote mainly knowledge transfer through practices that stimulate social interaction to share tacit knowledge in the project layer, increasing the risk of losing knowledge by keeping the knowledge localized inside a few individual’s minds. When it comes to coordination, practitioners utilize KBRs in their routines, through social collaboration within teams’ environment/settings. However, this process is nonsystematic, which brings inefficiency to KBR utilization resulting
in knowledge loss. It can also generate negative implications to the course of the software development, including meaningless searches in databases, frustration because of recurrent problems, and unawareness of knowledge sources. To support decision making related to knowledge retention, we have developed an initial version of the method to evaluate the criticality (KCEM) of a knowledge item, which is divided into two categories, relevance, and scarcity.

**Conclusion.** The evidence from the studies shows that ASD lacks on planning KM practices to manage the knowledge resources. This thesis contributes to filling this gap by offering the results of KM inefficiencies, and providing an initial tool to evaluate knowledge item. As a complement to this thesis, we have planned our long-term objective, which is to contribute to creating scalable KM solutions for companies adopting ASD. We divide this long-term objective into three studies: Carry out a complementary study to apply KCEM in different companies; explore efficient ways of storing codified knowledge in combination with the KCEM, and investigate how to define metrics to evaluate the outcomes of KM practices.

**Keywords:** Knowledge Management, Agile Software Development, Knowledge Resource
Acknowledgements

I express my gratitude to my supervisors, Krzysztof Wnuk, Tony Gorsheck, and Richard Berntsson Svensson for all insightful guidance and feedback thus far.

To my teaching mentor Mikael Svaehnberg for giving a handful of insights in this teaching journey.

To my colleagues at SERL Sweden for sharing this learning environment that is the doctoral studies. In particular, Javier Gonzalez-Huerta, Jefferson Molléri, and Bogdan Marculescu for the fun and relaxation moments.

To the friends that I found at BTH Ana Dallora, Thomas Sievert, and Vinicius Ludwig for sharing all the experiences of being a Ph.D. student.

To my dear husband João Felipe that is standing by me for the last 15 years with his unconditional support.

To my dear mother Joceli Andrade that guided and supported me through my entire life so that I could end up here with one more achievement.

To my father Edvaldo Barros that passed away ten months ago. He always celebrated my academic achievements and encouraged me always to go further. Unfortunately, he could not see this one, but I am pretty sure that today he would be happy.
Overview of Papers

Papers in this Thesis


Contribution Statement

**Raquel Ouriques** is the first author for all the papers that compose this Thesis. As lead author, Raquel was responsible for designing and conducting the
studies, collecting and analyzing data, and most reporting activities. In addition, she is the sole author of Chapter 1, the Introduction. The detailed authors’ contribution to the chapters are described next:

**Chapter 2:** Raquel Ouriques and Krzysztof Wnuk conceived the presented idea. Raquel Ouriques collected the data. Krzysztof Wnuk verified the analytical methods, and contributed to frame the findings of this work. Tony Gorschek and Richard Berntsson Svensson contributed to the discussion of the results and also to the final manuscript.

**Chapter 3:** Raquel Ouriques conceived the idea, conducted the writing and organization of of this vision paper. Krzysztof Wnuk, Tony Gorschek and Richard Berntsson Svensson contributed to the discussion of the results and also to the final manuscript.

**Chapter 4:** Raquel Ouriques conceived the idea and planning the overall research. All the authors provided feedback on the research design. Raquel Ouriques and Krzysztof Wnuk conducted the data collection. Krzysztof Wnuk verified the synthesis. Krzysztof Wnuk, Tony Gorschek and Richard Berntsson Svensson contributed to the discussion of the results and also to the final manuscript.

**Chapter 5:** Raquel Ouriques conceived the idea and designed the method. Ricardo Britto contributed with the research plan and gave insights on the method. João Felipe Ouriques contributed with data analysis. Krzysztof Wnuk and Tony Gorschek contributed to the discussion of the results and also to the final manuscript.

**Related Papers not in this Thesis**


**Funding**

The research in this thesis was funded by the Swedish Knowledge Foundation (KKS), the Federal University of Campina Grande (UFCG), and the Blekinge Institute of Technology (BTH).
Contents

Abstract vii

Acknowledgements ix

Overview of Publications xi
  Papers in this Thesis xi
  Related Papers not in this Thesis xii

1 Introduction 1
  1.1 Background and related work 2
  1.2 Research questions and contributions 6
  1.3 Research methods 9
  1.4 Summary of the results 11
  1.5 Future work 14

2 Knowledge management strategies and processes in agile software development: A systematic literature review 17
  2.1 Introduction 17
  2.2 Background 19
  2.3 Research methodology 25
  2.4 Studies Overview and Descriptive Statistics 30
  2.5 Results and Analysis 34
  2.6 Discussion and Research Gaps 44
  2.7 Threats to Validity 45
  2.8 Conclusions and Implications 46
  2.9 Appendix 48
## CONTENTS

3 Thinking Strategically About Knowledge Management in Agile Software Development 55
   3.1 Introduction .............................................. 55
   3.2 Knowledge Management in Strategic Management Domain . . 56
   3.3 Knowledge Management Strategies in Agile Software Development 58
   3.4 Conclusion ............................................. 61

4 Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources 63
   4.1 Introduction .............................................. 63
   4.2 Background and related work ................................ 65
   4.3 Research method .......................................... 69
   4.4 Findings: The Continuous Assimilation Model .............. 75
   4.5 Discussing implications for KM in ASD .................... 88
   4.6 Threats to validity ........................................ 92
   4.7 Conclusions .............................................. 94
   4.8 Appendix ................................................ 95

5 A Method to Evaluate Knowledge Resources in Agile Software Development 97
   5.1 Introduction .............................................. 97
   5.2 Background and Related Work ................................ 99
   5.3 Research Design .......................................... 100
   5.4 The knowledge criticality evaluation method - KCEM .......... 101
   5.5 Validation ................................................ 104
   5.6 Lessons Learned .......................................... 106
   5.7 Validity Threats .......................................... 108
   5.8 Conclusion .............................................. 109

References 111
List of Figures

1.1 Research methodology overview .............................................. 9

2.1 Automated search for start set selection. .............................. 28
2.2 Snowballing iterations. ...................................................... 29
2.3 Studies overview ............................................................ 31
2.4 Quality assessment .......................................................... 33
2.5 KMS - ASD ................................................................. 35
2.6 Distribution of KM practices in organizational layers through
   KMS-ASD ................................................................. 37

3.1 KMS-ASD [127] .............................................................. 57
3.2 Distribution of KM practices in organizational layers through
   KMS-ASD [127] .............................................................. 59

4.1 Research Process overview ................................................. 70
4.2 CHASM Model ............................................................... 76

5.1 Knowledge criticality ....................................................... 103
5.2 Rubrics results for the KITs ............................................. 105
5.3 Knowledge Criticality of the KITs evaluated ......................... 106
List of Tables

2.1 Keywords for automated search. .............................. 27
2.2 Data collection of primary studies. .......................... 30
2.3 Rigor scores .............................................. 52
2.4 Relevance scores .......................................... 53

4.1 Practitioners description ............................... 71

5.1 Rubrics for evaluating knowledge criticality .............. 102
Chapter 1

Introduction

Software development is undoubtedly a knowledge intensive activity [12,70], often associated with complex and intangible social resources that are difficult to reproduce, but give uniqueness to a company within an industry. Software development is also a dynamic activity where customer needs, market and technology trend fluctuate frequently and demand flexibility from software companies.

Many software companies have introduced Agile Software Development (ASD) methods as they are perceived to offer better flexibility and adaptability to changing environments. The ASD adoption positively impacts companies in several ways, e.g., coordination and communication, collaboration with customers, better task estimation, informal communication over written documentation, and focus on the work [57,156].

However, the increased flexibility that ASD offers comes with the cost of losing of the "big picture" of the product due to extensive focus on developing features, negative effects on learning, and low attention to architecture and design [7,36,57].

As a result, it becomes critical to manage Knowledge-Based Resources (KBRs), which are people's knowledge often related to specific skills such as technical, creative, and collaborative, which, in turn, regards to integration and coordination of multidisciplinary teamwork [115]. KBRs can be tacit, which is individual deriving from experiences, values and routines; and explicit, which is already systematized in formal language including guidelines, instructions or books [124]. The mismanagement of KBRs can lead to several negative effects. For example, barriers to collaboration and asynchronous communication in large companies that introduced ASD, competitive and knowledge loss [6,115].
Organizations manage their KBRs through codification and personalization strategies [78]. Codification relates to the activity of storing knowledge into artifacts (e.g., Wiki-based tools) when the knowledge nature tends to be explicit; personalization relates to socialization activities to communicate knowledge (e.g., team meetings and discussion groups) when the knowledge nature tends to be tacit.

As the software value chain is knowledge-based due to the high dependency on people [141], it becomes essential to understand what are the knowledge resources they own and how to utilize them to have a continuous readiness to quickly respond to the market by intrinsically changing and offering customer value [39,115].

This thesis focuses on supporting managing KBRs in software development by: (i) Understanding the current ASD environment regarding Knowledge Management (KM); (ii) Identifying knowledge-based resources in ASD and their implications for KM; and (iii) Providing and initial set of variables to evaluate knowledge criticality of knowledge items in ASD.

In this chapter, I describe the background on ASD and KM, the research questions and contributions. I summarize the research method utilized in each study; provide a synthesis of the results of each study, and future work. In chapter 2, I describe a systematic literature review on KM practices in ASD. In chapter 3, I discuss potential strategic management theories to tackle KM issues in ASD. In chapter 4, I describe the Continuous Assimilation Model (CHASM), which illustrates how knowledge-based resources support coordination in ASD. In chapter 5, I provide the initial validation of of our proposed Knowledge Critically Evaluation Method (KCEM) to evaluate knowledge items (KIT).

1.1 Background and related work

1.1.1 Agile software development

Agility and flexibility in responding to changing market and uncertainty [170] are focal points of Agile Software Development, which is an umbrella term that includes several methods and frameworks for guiding software development practices [14,39,57]. ASD prioritizes human factors such as interaction between people, and teamwork. Further, it focuses on delivering working software when new or updated requirements are expected even late in the development [14].

Dybå and Dingsøyr [57] analyzed the ASD literature and found several contradictory findings regarding the benefits of ASD. Some of the studies report
benefits and improvements regarding productivity, while others report the opposite. Benefits from the adoption of ASD were observed in relation to, e.g., collaboration with the customer, estimation, and focus on the work. On the other hand, limitations also were reported, such as not enough attention to architecture and design, and inefficiency of pair programming.

According to Conboy [39], ASD has a conceptual gap. To contribute on filling this gap, Conboy developed a taxonomy, based on the organizational and management literature, of agility in information system development. To contribute to agility, an agile method should at least facilitate one of the following: the creation of change, proaction in advance of change, reaction to change, and learning from change. Also, the agile method of choice should contribute to perceived economy, quality, or simplicity without poor performance in any of them.

In a recent study, Annosi et al. [7] observed that the time pressure imposed by the Scrum framework had negative implications for learning, and innovation. People keep focused on the tasks to develop at the cost of losing the big picture of the product. Li et al. [108] also discussed the negative effects of time pressure.

Dealing with constant changes is the central idea of ASD, and knowledge plays an essential role in this regard [141]. While ASD prioritizes informal communication due to the flexibility of the processes and reduced documentation, it relies to a large extent on knowledge that is shared among people [36].

Previous literature points out that companies who adopt ASD utilize mainly tacit knowledge [21], and rely on informal communication to share this knowledge between people in both co-located and distributed ASD [49, 69, 114]. Although tacit knowledge is recognized as a relevant resource for companies and crucial for innovation [122], it is only possible to be revealed when it is applied [73]. This knowledge is acquired through practices and transferred between people, which is costly, slow and uncertain [95].

However, in an industry where knowledge is the main competitive resource, the lack of management practices might jeopardize the application of knowledge in the production of goods and service. The benefits originated from KM could lead to a better competitive advantage when the knowledge has a crucial goal [55, 125, 141].

1.1.2 Knowledge management

A company’s competitive strategy, according to the resource based view of the firm [73], is substantially dependent on how the management maximizes the produced value, by allocating its resources and capabilities. These resources
are idiosyncratic and provide heterogeneity to a company in an market. A knowledge resource gets higher attention when it is involved in activities that require human interaction for production, inside and outside the companies, e.g., teams, and customers collaboration [9,149].

The knowledge-based view of the firm [73] derives from the resource-based view, and focuses on the people’s knowledge as the most strategical resource of a company. Knowledge-based resources might appear as specific skills such as technical, creative, and collaborative, which, in turn, regards to integration and coordination of multidisciplinary teamwork [115].

The challenge of logically allocating the knowledge resource is related to the dilemma of extracting and using knowledge that is distributed in many individuals’ minds [80]. Although knowledge has its own theoretical definition, it is commonly used interchangeably with information. According to Nonaka [124], information is a collection of a particular arrangement represented by a flow of messages, whereas knowledge is the meaning created by the combination of information and individuals beliefs.

Knowledge is context dependent, dynamic and created through social interactions between individuals. Knowledge is divided in tacit – rooted in individual’s mind and it is a result of values, beliefs, life experiences, emotions, procedures, actions and routines; and explicit – it is easy to transmit since it is already systematized in data, formula, manual, books, specifications, along with others [79,124].

Even though knowledge is considered as the primary source of sustainable advantage for several companies [138], managing it remains challenging. Different fields have studied knowledge Management (KM) due to its relevance and interdisciplinary nature and proposed many concepts, theories, and applications. The term started to appear in academic publications in 1986 [171].

In this thesis, I adopt the definition of knowledge management (KM) as the effort to manage organizations’ workforce through information and communication technologies or creation of a corporate culture that focuses on social processes that facilitates the sharing between individuals, aiming to reach a sustainable source of advantage [83,165]. KM strategy is the effort to formulate plans for actively managing knowledge [157]. The strategies can be codification and/or personalization, rooted in the company’s corporate strategy [24]. Codification involves the storage of knowledge into databases for further use (e.g., wiki-based tools [88], while personalization focus on human interaction to communicate knowledge (e.g., communities of practice) [78].
1.1 Background and related work

The KM strategies promote a set of knowledge processes that represent individual’s cognitive, social practices and the culture that shapes knowledge in organizations [2]. They are four KPs:

- Knowledge Creation (KC) comprises the development of new ideas, concepts or knowledge replacement by the constant combination between tacit and explicit. A critical step to trigger this process is social interaction, which enables individuals to share and develop new knowledge [122].

- Knowledge Storage and retrieval (KS) is related to the organizational memory, how it keeps the knowledge through documentation, databases, networks of individuals and so on. This memory is built with past experiences, events, and procedures that affect the organization’s current activities.

- Knowledge Transfer/sharing (KT) refers to transfer knowledge to areas needed within the organization. The knowledge disperses in different levels, between groups, individuals, across groups, and from the group to the organization. In this process, the main challenge is to know what knowledge the organization need to share, because usually, they do not know what they know. In ASD context, rotating individuals facilitate the transfer of knowledge between team members [28].

- Knowledge Application (KA) indicates the use of the knowledge as a competitive advantage through improvements in organization capability. Three mechanisms can integrate the knowledge: directives – the collection of rules and procedures; organizational routines – regards to coordination patterns and tasks development; and self-contained task – created especially for problem-solving, when there is no support from directives or organizational routines.

Managing knowledge resources increases the return and sustains the advantages of new, transferred, and applied knowledge [47,165]. In any organization, knowing the knowledge needs is one of the main steps for an effective KM, and also understand that filling repositories with knowledge is not the best approach to have favorable outcomes [107].

According to Robillard [138], there is a lack of understanding of how knowledge is processed in cognitive structures. Additionally, this process cannot be entirely automated. Different strategies are adopted aiming to codify the knowledge to automate managing it. On the other hand, several strategies focus on
stimulating the cognitive structure of individuals through social interactions (personalization).

Depending on the organizational goals, a combination of these strategies may enhance one or more KPs. Considering that software development is a knowledge-intensive activity [10], that combines tacit and explicit knowledge during its lifecycle, it is valuable to understand how these constructs are inter-related to apply a good strategy combination.

1.2 Research questions and contributions

This thesis contributes to the software engineering field by providing another perspective on directions that KM should take to improve knowledge resource management in ASD.

This section outlines the research questions (RQ) that steered this thesis, and the research contributions.

1.2.1 Research questions

Three research questions are investigated in this thesis:

RQ1. How companies adopting ASD implement KM strategies?

Knowledge-intensive software companies that adopt ASD should rely on efficient implementation of KM strategies to promote different KPs to gain competitive advantage. There are several studies that focus on practices for specific KPs. Although the research literature in ASD encompass several studies on practices for specific KPs, there is a little understanding how KM strategies relate to the different KPs. Thus, we contribute filling this gap by means of focusing on how companies adopting ASD manage the knowledge asset implementing KM strategies, focusing on the KPs that they promote in the different organizational layers.

To answer this research question, we explored empirical studies in the literature to understand how companies that adopt ASD implement KM strategies utilizing practices that promote the KPs in the different organizational layers. The results provide increased comprehension concerning how the companies implement KM strategies with practices that promote the KPs.
1.2 Research questions and contributions

RQ2. How knowledge-based resources support coordination in agile software development?

Agile methodologies have given teams more autonomy regarding planning tasks and executing them. As a result, coordination gets more flexible, but much relevant knowledge remains undocumented and inside teams’ borders, due to informal communication and reduced development documentation.

As the software value chain is knowledge-based and highly dependent on people, the lack of practice to manage knowledge as a resource might jeopardize its application in software development. The resource-based view of the firm provides a different perspective on the utilization of knowledge, assisting the identification of the Knowledge-Based Resources (KBRs) that allow a company to have a continuous readiness to quickly respond to the market changes.

To answer this question, I based myself on the strategic management field that consider knowledge as manageable resource (see chapter 3), and collected data from practitioners from different companies to group in the Continuous Assimilation Model (CHASM) (see chapter 4). The findings contribute to the main objective of this thesis by exploring how the KBRs support coordination in Agile Software Development (ASD), and also the inefficiencies resulted of knowledge resources mismanagement.

RQ3. How knowledge management could improve in ASD?

Companies use different Knowledge Management (KM) practices to retain and share knowledge. Knowledge retention practices relates to the preservation of people’s knowledge that is crucial to an organization to perform its primary functions. However, it is often the case that knowledge retention is carried out in an ad-hoc way.

To answer this question, I investigate how to improve knowledge retention practices by defining how critical knowledge resources are, and improve knowledge retention practices (see chapter 5).

The findings contribute to the main objective of this thesis by encouraging a change on the way we perceived knowledge towards a resource and reducing the level of abstraction of the topic.
1.2.2 Research contributions

This thesis contribute to the software engineering field by incorporating strategic management for managing knowledge as a resource in agile software development contexts. Also, we provide an initial method to support companies decision making relate to knowledge retention. The detailed contributions are:

C1. **Synthesis of the empirical software engineering literature to understand how companies adopting ASD manage the knowledge resource implementing KM strategies (RQ1).** We identified the KM practices and classified them according to the knowledge process they promoted. The results contribute to summarize and analyze the state-of-the-art of the literature in relation to KM, its application and relation to ASD; illustrate the KM strategies in the hierarchical layers of software development companies that adopt ASD, through a conceptual classification framework; and, provide comprehension concerning how the companies implement KM strategies with practices that promote the KPs.

C2. **Introduce another perspective of knowledge (RQ2).** We utilized the resource-based view of the firm [11] to support identifying relevant knowledge resources. Several companies perceive the strategic relevance of knowledge as the main competitive resource, especially in knowledge-intensive environments. However, building and maintaining the competitive advantage based on the knowledge resource, requires a strategic plan that integrates the long-term vision with the knowledge resources [75]. Our discussion contributes to a change on the way knowledge is perceived in ASD contexts. It encourages the use of knowledge as a resource to support in changing environments, where companies need to adapt themselves in order to be more responsive and monitor current and future knowledge needs.

C3. **Understanding of the role of knowledge-based resources in ASD coordination (RQ2).** We examined the knowledge from the resource-based view perspective, which gave us insights about how this resource supports coordination in ASD. The results provide the following contributions: An empirical investigation to identify the knowledge-based resources in agile software development; The development of the Continuous Assimilation Model (CHASM), which illustrates how knowledge-based resources support coordination in ASD; and, a discussion on possible implications for future research and potential solutions for managing the knowledge resources.
1.3 Research methods

C4. Provide a method to support companies to systematically retain knowledge in ASD (RQ3). We proposed the version n.01 of the Knowledge Critically Evaluation Method - KCEM, which aims to support companies to systematically retain knowledge in ASD contexts. It also contributes to avoid the accumulation of unnecessary artifacts. We designed KCEM to be lightweight and easily applicable by practitioners.

1.3 Research methods

We used a mixed-methods approach [139] to address the objective of this thesis (see figure 1.1), and answer the research questions aforementioned in section 1.2.1. The methods selected to conduct the studies include systematic literature review, grounded theory, and improvement case study. A brief description of the method and rationale for choosing them are described further.

![Figure 1.1: Research methodology overview](image)
Systematic literature review

We applied Systematic Literature Review (SLR) as the first method to support achieving our main objective by synthesizing the empirical evidence regarding KM in ASD.

In chapter 2, we detailed the research protocol. We followed the guidelines proposed by Kitchenham [93] to structure the research question, search strategy, study selection, exclusion criteria, data extraction, and synthesis. Besides, we applied the Rigor and Industrial Relevance Model proposed by Ivarsson and Gorschek [86] to perform the quality assessment of the primary studies.

We reported the identified main gaps related to each KP. Also, the general patterns in the findings. In chapter 3 we extend the discussion we have initiated in the previous chapter by adding an organizational theory which we believe can address some of the identified gaps.

Grounded theory

Grounded theory method is useful to explore areas that there is a lack of explanation or description [139]. We chose to conduct a qualitative study to efficiently capture data regarding the complex phenomena of the human behavior specially linked to the abstract concept of knowledge [85,147].

We investigated the planning and coordinating activities utilized by practitioners in software development companies that adopt ASD. We focused on these activities because the coordination mechanisms stimulate the emergence of conditions where people are able to combine their specialized knowledge [73]. We collected the data over interviews and documents (e.g., quality framework and coordination scheme). We examined the data through open coding (text and audio) to create the concepts that are part of and explain the categories established [147]. In chapter 4, we detail the research design and data analysis procedures.

Improvement case study

Based on the grounded theory study (see chapter 4), as well as on the output from a preliminary seminar with a company, we created the categories and rubrics for analyzing knowledge criticality and the strategy for the classification.

We conducted an improvement case study [140], which we have detailed in chapter 5 to develop and evaluate KCEM. The research follows the guidelines for technology transfer between industry and academia proposed by Gorschek
et al. [72]. The case and unit of analysis is Ericsson, a Swedish company that develops telecommunication solutions.

1.4 Summary of the results

In this section I describe the main results obtained through the methods described in the aforementioned section 1.3, and how they answer our research questions.

1.4.1 Knowledge management in agile software development

We investigated how software development companies adopting ASD implement KM strategies through practices that promote the KPs in the organizational layers (RQ1), illustrated in the conceptual classification framework described in chapter 2. Then, we list the KM practices reported in the primary studies regarding the KM strategy they implement, and mapped them into the organizational layers. In each layer, we classified the practices regarding the type of KM strategy they implement, and which KP they promote.

The results point that KM strategies in ASD promote mainly the knowledge transfer process with practices that stimulates social interaction to informally sharing tacit knowledge in the project layer. However, there is also a risk of losing knowledge or not transferring it. The result is that knowledge might get localized inside a few individual’s minds instead of being propagated.

Another relevant aspect of the results relate to the knowledge storage process. We notice that software companies still struggle to know what knowledge they should store. More important than having a tool, is to know what to store regarding information or knowledge.

Treating knowledge as a resource implies that it needs to be managed with a logical and structured approach [11, 141]. To contribute to this discussion, we retrieve KM aspects from strategic management domain (see chapter 3) to steer this research on formulating KM solutions (RQ2).

1.4.2 Knowledge-based resources

Knowledge is recognized as a major resource for software development. However, the lack of understanding about how to manage this abstract concept in real practices might hinder its utilization.
We examined the knowledge from the resource-based view perspective ($RQ_2$), which gave us insights about how this resource supports coordination in ASD.

We gather the results in the Continuous Assimilation Model (CHASM), which represents the (KBRs) supporting coordination in agile software development environments. CHASM comprises 44 KBRs (RQ1) that are classified in six categories:

- Business analytic perspective
- Product systemic reasoning
- Social collaboration
- Ability to codify and transmit knowledge
- Team environment/settings
- Inefficient utilization of the knowledge resource

The overall results show that the practitioners utilize KBRs in their routines, through social collaboration within teams environment/settings. However, this process is non systematic, which brings inefficiency in terms of knowledge resource utilization resulting in knowledge loss. It can generate negative implications to the course of the software development including meaningless searches in databases, frustration because of recurrent problems, redesign of solutions, and unawareness of knowledge sources.

To utilize the model, it is important that the practitioners notice that similar to any other type of resource, the knowledge-based ones have different levels of importance to each software company. As a starting point, we suggest that practitioners prioritize critical KBRs, which we believe will be tightly connected to the corporate strategy, and develop strategies to manage them.

### 1.4.3 Towards knowledge resource management in agile software engineering

To support decision making related to knowledge retention, we have developed an initial version of the KCEM, which is a method to evaluate the criticality of a knowledge item (see chapter 5).

We define knowledge criticality as the degree to which a knowledge item is essential to what it has been applied. The method evaluates Knowledge Items (KIT), which we define as the particular unit of knowledge. A KIT can be
different things such as a process and its associated knowledge. A KIT can be either tacit or explicit, e.g., coordination and interpersonal skills or knowledge stored by developers in wikis.

Criticality is divided into two categories, relevance and scarcity, which we define as follows:

- **Relevance** - potential for satisfying a particular need, e.g., within a team or larger agile setups.
- **Scarcity** - relates to the extent that a KIT is readily obtainable, e.g., the number of experts that has a specific skill.

Each category has a set of rubrics, which can be used by an individual or by a group. The result of the calculated scores for each rubric is visualized in a 2-dimensional chart, which has four quadrants for recommendation categories for managing the evaluated knowledge. The quadrants are described as follows:

- **Q1 - Slight criticality:** The KIT has low scores for both scarcity and utility, i.e., it has low complexity and probably will be readily available when needed.

- **Q2 - Low criticality - Vigilant:** The KIT’s utility is low, but high scored in scarcity, which might be challenging to acquire it quickly. It might not have, e.g., homogeneous importance across teams or to a development process. Thus, the recommendation for this knowledge item is to make efforts to, e.g., duplicate or make it redundant on occasions where and when it is needed.

- **Q3 - Medium criticality - Handy:** The knowledge item is mostly available and important. The recommendation is to facilitate accessibility through efficient search processes, either through socialization or codification processes. For example, codify knowledge in artifacts (instructions, guidelines) or, in case of tacit knowledge, make the source visible.

- **Q4 - High criticality - Strategic:** The knowledge item is crucial and potentially contributes to continuous readiness to absorb unforeseen changes in agile contexts. Thus, it is strongly recommended to establish knowledge retention practices for this knowledge item.

We have conducted a preliminary validation of the method, which was divided in two parts: lab validation and static validation. Both validations show promising results (described in chapter 5).
Considering that the main goal of our research is to support software-intensive companies, the results of the static validation are of special interest. The practitioners who used our method highlighted three main advantages: easy to understand and use, provide a different perspective on a KIT by visualizing the criticality chart, and reduce the level of abstraction associated to a knowledge subject area.

The findings of this study have important implications for future KM practices including changing the way we perceived knowledge towards a resource and reducing the level of abstraction of the topic.

1.4.4 Limitations

Besides the detailed threats to validity in chapters 2, 4, and 5, we recognize that this thesis has limitations.

I would like to mention that although we introduce strategic management perspective for managing knowledge in ASD environments, the connection between the two areas is not deeply built.

Furthermore, the KCEM method was evaluated in a single company, which hinder its applicability to other ASD contexts. However, an improved version of the method is envisaged (see section 1.5).

1.5 Future work

The long term objective of this thesis is to contribute on creating a scalable KM solutions for companies adopting ASD. As the next steps, we plan the following:

- Carry out a complementary study to apply KCEM in different companies. Our aim with this study is to adjust the method considering the various companies contexts by collecting feedback from practitioners, and observing the contexts variations.

- A natural progression of this work is also to explore efficient ways of storing codified knowledge in combination with the KCEM. This idea is partially steered by the emerged results shown in chapter 4, which discuss the lack of trust in the knowledge resource that is codified in artifacts due to outdated, and poor structure.

- We intend to investigate how to define metrics to evaluate the outcomes of KM practices. The lack of effectiveness measures for KM practices is
1.5 Future work

still a gap that we identified and also pointed by other researchers (see chapter 3).

• We also plan to explore what indicates the obsolescence of a knowledge item that has been codified.
Introduction
Chapter 2

Knowledge management strategies and processes in agile software development: A systematic literature review

This chapter is based on the following paper:


2.1 Introduction

Software-intensive product development companies struggle to stay competitive due to fierce competition and increased pressure, forcing them to quickly release new products to the market [118,137]. These forces push companies to improve resource management aiming for better product quality, creativity or efficient
development process [47]. Many companies have also introduced Agile Software Development (ASD) methods, as they are perceived to offer better response to frequently changing market needs, more flexible software development methods and shorter learning cycles [39].

The increased flexibility that ASD promises comes with the cost of prioritizing informal communication between team members over written documentation [36] or loss of the “big picture” of the product due to extensive focus on developing features [7]. As a result, managing the knowledge asset becomes critical, and the lack of it can lead to several negative effects. For example, barriers to collaboration and asynchronous communication in large companies that introduced ASD [6], and competitiveness loss [115].

Mastering how to manage knowledge as a competitive asset [2] can improve team communication and the change responsiveness. The utilization of Knowledge Management (KM) strategies can significantly support achieving and sustaining competitive advantage [83,165] and brings several benefits to software development [67,141], e.g. that effective networks for tacit knowledge sharing in ASD contribute to continuous process improvement [22].

How to manage knowledge in ASD is still not well understood or investigated [21]. Rus and Lindvall [141] affirm that the reason for failures in KM deployments happens because many companies do not establish their goals in the KM strategy, and manage documents instead of knowledge.

KM focuses on the effort to manage an organizations’ workforce through social processes that facilitate interaction between individuals [83]. KM also has established principles for dynamic activities that focus on the Knowledge Processes (KPs): creation, storage/retrieval, transfer, and application [2].

Knowledge processes often use practices that implement codification and/or personalization strategies; which should be rooted in the company’s corporate strategy [24], for example, physical open spaces, which implement a personalization strategy, could promote knowledge transfer between individuals in physical structure that facilitates the working together [145]. Although the research literature in ASD encompass several studies on practices for specific KPs [24,145]; what is lacking, however, is understanding how KM strategies relate to the different KPs in companies adopting ASD.

While the software engineering field had addressed the storage and retrieval knowledge process, the field is still distant from the KM mainstream research, and processes such as knowledge creation, transfer and application remain largely unexplored [21]. This study contributes to filling this gap by means of focusing on how companies adopting ASD manage the knowledge asset implementing KM
strategies, focusing on the KPs that they promote in the different organizational layers.

The contribution of the study is threefold: 1) Summarize and analyze the state-of-the-art of the literature in relation to KM, its application and relation to ASD, 2) Illustrate the KM strategies in the hierarchical layers of software development companies that adopt ASD, through a conceptual classification framework, and 3) Provide comprehension concerning how the companies implement KM strategies with practices that promote the KPs.

The rest of this paper is organized as follows: Section 2.2 introduces the background on ASD, KP, KM strategies and the conceptual classification framework. Section 2.3 describes the procedure for conducting this systematic literature review (SLR). Sections 2.4 2.5, and 2.6 report the results, analysis, and discussions. In Section 2.7, we discuss threats to validity, and in Section 2.8 we provide conclusions and implications.

2.2 Background

In this section, we shortly present basic concepts of KM, explaining the types of knowledge, the knowledge processes, and the KM strategies. We provide a brief overview of ASD and a summary of the previous literature reviews in the area. We also compare them to our study, explaining the major differences.

2.2.1 Knowledge management

The challenge of allocating the knowledge asset logically is related to the dilemma of extracting and using knowledge that are distributed in many individuals’ minds [80]. Although knowledge has its own theoretical definition, it is commonly used interchangeably with information. According to Nonaka [122], information is a collection of a particular arrangement represented by a flow of messages, whereas knowledge is the meaning created by the combination of information and individuals beliefs.

Knowledge is context dependent, dynamic and created through social interactions between individuals. Knowledge is divided in tacit – rooted in individual’s mind and it is a result of values, beliefs, life experiences, emotions, procedures, actions and routines; and explicit – it is easy to transmit since it is already systematized in data, formula, manual, books, specifications, along with others [79,124].
Even though knowledge is considered as the primary source of sustainable advantage for several companies [47], managing it remains challenging. Different fields have studied knowledge Management (KM) due to its relevance and interdisciplinary nature and proposed many concepts, theories, and applications. The term started to appear in academic publications in 1986 [171].

In this study, we adopt the definition of knowledge management (KM) as the effort to manage organizations’ workforce through information and communication technologies or creation of a corporate culture that focuses on social processes that facilitates the sharing between individuals, aiming to reach a sustainable source of advantage [83,165].

KM strategy is the effort to formulate plans for actively manage knowledge [157]. The strategies can be codification and/or personalization strategies, which are rooted on the company’s corporate strategy [24]. Codification involves the storage of knowledge into databases for further use (e.g., wiki-based tools [88]), while personalization focus on human interaction to communicate knowledge (e.g., communities of practice) [78]. The KM strategies promote the set of knowledge processes that represent individual’s cognitive, social practices and the culture that shapes knowledge in organizations [2]. They are four KPs:

- **Knowledge Creation (KC)** comprises the development of new ideas, concepts or knowledge replacement by the constant combination between tacit and explicit. A critical step to trigger this process is social interaction, which enables individuals to share and develop new knowledge [122]. In ASD, we could exemplify this process being triggered by pair programming, where developers elaborate create new knowledge to complex problems solution based on interaction;

- **Knowledge Storage and retrieval (KS)** is related to the organizational memory, how it keeps the knowledge through documentation, databases, networks of individuals and so on. This memory is built with past experiences, events, and procedures that affect the organization’s current activities. For example, inside agile teams, it is common utilize wiki-based tools to store additional documentation related to meetings and problems solution developed by the team. In SCRUM framework, retrospective meetings could support KS;

- **Knowledge Transfer/sharing (KT)** refers to transfer knowledge to areas needed within the organization. The knowledge disperses in different levels, between groups, individuals, across groups, and from the group to the organization. In this process, the main challenge is to know what
knowledge the organization need to share, because usually, they do not know what they know. In ASD context, rotating individuals facilitate the transfer of knowledge between team members [23]. Wiki-based tools is another example of transferring knowledge about the software being developed from agile teams to different departments in a company [56];

• *Knowledge Application* (KA) indicates the use of the knowledge as a competitive advantage through improvements in organization capability. Three mechanisms can integrate the knowledge: directives – the collection of rules and procedures; organizational routines – regards to coordination patterns and tasks development; and self-contained task – created especially for problem-solving, when there is no support from directives or organizational routines. In software development companies that adopt SCRUM framework, review sprint and retrospectives might be a good support for KA activities by effectively applying knowledge gained from previous sprints and projects.

A knowledge asset is a knowledge-based resource, for example, individual or team solutions developed for complex problems, related to feature development, based on individuals' previous knowledge and experience. Managing knowledge assets increases the return and also keeps the continuing advantages of new generated, transferred and applied knowledge [2,47]. In any organization, knowing the knowledge needs is one of the main steps for an effective KM, and also understand that filling repositories with knowledge is not the best approach to have favorable outcomes [107].

To process and apply knowledge is an intuitive activity, derived from mental mechanisms that are capable of connecting and merging conflicting knowledge [122,124]. Since knowledge is a large part of the resources used to build software products, it is crucial to know the cognitive structures process knowledge.

According to Robillard [138], there is a lack of understanding of how knowledge is processed in cognitive structures. Additionally, this process cannot be entirely automated. Different strategies are adopted aiming to codify the knowledge to automate managing it. On the other hand, several strategies focus on stimulating the cognitive structure of individuals through social interactions (personalization). Depending on the organizational goals, a combination of these strategies may enhance one or more KPs. Considering that software development is a knowledge-intensive activity [10], that combines tacit and explicit knowledge during its lifecycle, it is valuable to understand how these constructs are interrelated to apply a good strategy combination.
2.2.2 Agile software development

Traditional software development processes struggle to offer the required flexibility and change responsiveness [82]. Promptly responding to changes is the prevalent idea of ASD [14] that had its principles disseminated in 2002. ASD principles arise as a contradiction to the traditional software development which is inspired by the project cycle and guided by tasks and documents [119]. Instead of extensive documentation to guide the product development, the ASD relies on the team capability of communicating, collaboration, and being creative [47].

Several ASD methods were suggested based on the principles established in 2001. Dybå and Dingsøyr [57] point out six primary agile methods: Crystal methodologies; Dynamic software development method (DSDM); Feature-driven development, Lean software development, Scrum, and Extreme programming (XP; XP2).

Successfully adopting ASD requires several adaptations, since it highly depends on the context that is applied. Contextual factors such as system size, customer demand for software, team expertise, and highly constrained quality attributes requires evaluation in agile adoptions [97]. A flexible organizational structure is also required for agile software development, where collaboration takes place.

Agile software development methods have been adopted in small collocated teams and also in large companies that have distributed teams [31, 109, 131]. Studies had already reported success on this adoption, although several challenges remain. Those challenges are related to team coordination, requirements definition, communication and knowledge sharing [102, 130].

Traditional software development focuses mainly on explicit knowledge while in the agile context, the knowledge management activities are focused mainly on tacit knowledge [21]. Tacit knowledge is difficult to articulate since it is embodied in peoples’ mind; it is a result of values, beliefs, and experiences [122]. In addition to this complexity, there are two notable obstacles for managing knowledge asset in software companies: time to transform the tacit knowledge into explicit and use it, and time pressure faced by project managers to deliver running code. However, benefits originated from KM could lead to a better competitive advantage when the knowledge has a crucial goal [55, 125, 141].

2.2.3 Previous systematic literature reviews

We identified five literature reviews that explore knowledge management and software development.
Bjørnson and Dingsøyr published a review [80] involving knowledge management in software engineering in 2008. They reported empirical studies on knowledge management in software engineering, including knowledge management concepts, major findings, and research methods. They observed that tacit knowledge is the focus of knowledge management activities in ASD, while explicit knowledge is more addressed in traditional development processes. They noticed that there were more lessons learned studies (or industrial case studies) than scientific contributions. Recommendations from the authors suggest that it is necessary to explore how to manage tacit knowledge.

In 2013, Camacho et al. published a systematic literature review [30] focusing on research on knowledge transfer in software engineering. They explore how to imply, how to measure, and how it occurs in ASD. Knowledge transfer is performed mainly through technology, using repositories to store information about the work that done and tools to increase collaboration between distributed sites. According to their findings, the conventional measures relates to the amount of innovation and number of new concepts successfully generated. It can also be measured with different rubrics, such as sales, customer satisfaction, number of customers, the value created by research and development activities, and return investment. They claim that the intensive communication facilitates knowledge transfer in the agile environment due to less documentation generation, the focus is on the interaction between people instead of processes and tools.

In 2014, Cabral et al. conducted a systematic literature review [176] about knowledge management in agile methodologies. They considered empirical and non-empirical studies in their analysis. They identified seven topics in the selected papers: project documentation, transfer and collaboration of tacit knowledge, adoption of knowledge management methodology, tools for knowledge management, human and social factor, use of communities of practice, and knowledge artifacts and experience knowledge. Regarding the software development cycle, the authors suggest that adopting knowledge management practices not only change the development cycle but also increase its costs. Most of the tools for knowledge management focus on communication between participants, which, for them, implies on the need for more studies where tools are more focused on the software development process.

In 2015, Ghobadi conducted a systematic mapping study, to understand what drives knowledge sharing in software development teams [66]. She analyzed 49 papers and extracted 44 knowledge sharing drivers that were classified in four major categories (people-related, structure-related, task-related, and technology related), which had seven sub-categories (diversity-related drivers, capability-related drivers, team perception drivers, team organization drivers,
team organization drivers, organizational practices drivers, task-related drivers, and technology drivers). The analysis suggests that the most cited drivers are the ones related to team perception, which refers to values, perception, and attitudes that drive the team to share; organizational practices, that refers to communication networks, organizational norms and practices; and technology-related, referring to templates, tools, and methodologies.

In 2017, Zahedi et al. published their review [177] about knowledge sharing challenges and practices in global software development. They grouped challenges and practices into six categories: management, team structure, work processes/practices, team cognition, social attributes, and tools. Most of the reported challenges regard the work processes, followed by social attributes and technology/techniques categories. The study reveals that one of the significant challenges faced by global software development is to deal with tacit knowledge and that there are costs involved in knowledge sharing, such as travel between sites and keep documentation up to date, that may not appear on the project planning. Among the reported practices, temporary collocation is the most popular knowledge practice used by the companies. Other practices are also reported, such as groupware tools to enable frequent communication and build social ties. In general, the majority of companies tend to influence the team's social potential to share knowledge.

Comparison between this study and the existing related reviews

From the five SLR’s that we identified, three of them investigate knowledge sharing/transfer in different contexts of software development. One of them explores knowledge management in software engineering. The closest SLR to our work is the one reported by Cabral et al. [176] that aims to investigate the main topics of knowledge management in ASD. On a high-level, it seems similar. However, in detail, several differences can be perceived between this study and the other SLRs, regarding the following aspects:

- **Research questions.** The research questions elaborated in this study are different from all previous SLR’s, (see section 4.3). It may have an overlap of findings when comparing specific knowledge management processes such as knowledge sharing since the study published by Zahedi et al. [177] focus on challenges and practices of this KP. However, they consider this aspect in global software development. Our study explores all KPs in companies that adopt ASD.
2.3 Research methodology

- **Year.** Our search method, including automated search and snowballing procedure, was carried out on April 2018, while the closest to our study, reported by Cabral et al. [176], was performed in 2009, portraying a difference of eight years and 11 months. We consider this is a reasonable timespan, considering that software development area has rapidly changed over the years, e.g., more companies are introducing agile methods, and several of them are also adopting it in global software development.

- **Search string.** The search string used in this study differs since did not adopt composite words for knowledge management and combine the words for agile software development differently. We also used different databases to run the search strings for the automated search; more is included, for example, Web of Science, that has better coverage in journals of high impact [1].

- **Search method.** Our study adopted two search methods, automated search and snowballing, while the other studies adopted mainly automated search. The automated search was used to find the start set for the snowballing procedure.

- **Inclusion and exclusion criteria.** We defined different inclusion/exclusion criteria, particularly the one related to empirical studies, through which we could aggregate different findings.

- **Quality assessment.** Since we were looking for empirical studies, we adopted the Rigor and Relevance model [86] to classify and evaluate the studies based on the rubrics for rigor and relevance to better judge the empirical level of the studies selected for analysis in our study.

Most importantly, to the best of our knowledge, this is the first study that focuses on KM processes in ASD and their association with KM strategies. Previous studies target different aspects of KM in ASD. Therefore, we believe this review contributes to a better understanding of how companies that adopt ASD manage the knowledge asset. These findings also highlight the role of an integrated approach to KM that consider not only the project level, but how these practices should be related to the company’s overall goals.

### 2.3 Research methodology

We applied SLR to synthesize the empirical evidence regarding KM in ASD. In the subsections that follow we present the subsequent steps: protocol develop-
Knowledge management strategies and processes in agile software
development: A systematic literature review

2.3.1 Protocol development

We followed the guidelines proposed by Kitchenham [93] to structure the research question, search strategy, study selection, exclusion criteria, data extraction, and synthesis. Besides, we applied the Rigor and Industrial Relevance Model proposed by Ivarsson and Gorschek [86] to perform the quality assessment of the primary studies.

2.3.2 Research question

We defined the research questions:

RQ1: How do knowledge management strategies promote knowledge processes in ASD?

RQ2: To what degree have knowledge management strategies been validated in industrial settings?

RQ3: How have knowledge management practices been distributed in companies that adopted ASD?

Motivation: Previous studies have focused on specific KPs, e.g., knowledge sharing [66,177], and related personalization strategies with knowledge sharing practices. However, the rest of the KPs (creation, storage, and application) remains uncovered regarding its practices and their relation to the KM strategy. As a consequence, our motivation relies on the need to gain comprehension about how software development companies adopting ASD implement KM strategies through practices that promote the KPs in the organizational layers, illustrated in the conceptual classification framework (see section 2.5).

2.3.3 Search process and study selection

The search process followed two approaches. According to Kitchenhan et al., [93] and Wohlin [173], combining different approaches is a way of having the best possible literature coverage. We applied automated search and snowballing to select the primary studies.

The definition of the start set for the snowballing [173] procedure was carried out with the help of a database search. We searched for essential keywords and common terms, synonyms, and abbreviations in previous literature reviews [21,57,176]. Keywords related to agile software development were derived from
Table 2.1: Keywords for automated search.

<table>
<thead>
<tr>
<th>Agile Software Development - (ASD)</th>
<th>Knowledge Management - (KM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 agile AND software</td>
<td>11 knowledge</td>
</tr>
<tr>
<td>2 “extreme programming”</td>
<td>12 learn*</td>
</tr>
<tr>
<td>3 xp AND software</td>
<td></td>
</tr>
<tr>
<td>4 scrum AND software</td>
<td></td>
</tr>
<tr>
<td>5 crystal AND software AND (clear OR orange OR red OR blue)</td>
<td></td>
</tr>
<tr>
<td>6 dsdm AND software</td>
<td></td>
</tr>
<tr>
<td>7 “feature driven development”</td>
<td></td>
</tr>
<tr>
<td>8 fdd AND software</td>
<td></td>
</tr>
<tr>
<td>9 lean AND software</td>
<td></td>
</tr>
<tr>
<td>10 “dynamic system development method”</td>
<td></td>
</tr>
</tbody>
</table>

Dybå and Dingsøyr [57], while keywords related to knowledge processes were not limited to composite words, for example, "knowledge creation" or "knowledge sharing". We decided to use only knowledge and learn variations (e.g., learning), due to the lack of information about the existence of studies that did not use combined words. By doing so, we reduced the precision of the search string, elevating the number of papers found (4951). Although, adopting this strategy we reduced the risk of missing relevant papers.

We combined all relevant keywords (refer to table 2.1) using Boolean operators, and the search string result was:

\[(1 \text{ OR } 2 \text{ OR } 3 \text{ OR } 4 \text{ OR } 5 \text{ OR } 6 \text{ OR } 7 \text{ OR } 8 \text{ OR } 9 \text{ OR } 10) \text{ AND } (11 \text{ OR } 12)\]

The search string execution and paper selection and screening were conducted in April 2018. We executed our search string in Engineering Village that focus on engineering databases; Scopus, that is considered the most extensive citation and abstract databases; and ISI Web of Science, that has lower coverage than Scopus, but the journals covered has a higher impact [1].

**Automated search**

Fig. 2.1 presents the start set identification process (A1) on the databases (Scopus, Web of Science and Engineering Village) search was 4951. The paper’s metadata retrieved from databases were stored using Zotero. A total of 1019 duplicates were removed, which left 3572 papers for further analysis.

At stage 2, the first author excluded papers not written in English and studies that did not undergo peer-review, resulting in 3222 candidates. On the
third stage of the selection process, the first author excluded the papers that do not consider KM in ASD (exclusion criterion 3), resulting in 101 papers.

Several papers did not present enough information that allowed us to exclude them by title or abstract. Because of that, the first author, together with the second author (Stage 4 – A1 A2) screened the remaining 101 papers and excluded 71 papers based on the third criterion.

At stage 5, the first and the second authors individually excluded papers based on criterion 4 – Non-empirical papers. The number of papers resulted from this stage was 24. In this stage, we calculated the Cohen’s Kappa coefficient of agreement [37] to measure the degree of agreement between the researchers. The coefficient value was 0.87, which represents strong agreement [100].

**Snowballing iteration 1**

We performed backward and forward snowballing on the remaining 24 papers. Fig. 2.2 shows the process and the number of papers candidate for selection. We collected 2043 papers in the first snowballing iteration.

We first removed all papers that we already examined on the automated search. After that, by reading the titles and occasionally the abstracts, the first author selected 35 candidate papers. The first and the second authors individually screened the papers applying the third exclusion criteria, as mentioned
earlier, to the 35 candidate papers. The Cohen’s Kappa coefficient of agreement in this stage was 0.18, indicating a slight strength of agreement.

In several cases, it was not clear whether the context of the study was the agile environment, which explains the weak agreement. Due to the lack of information and to not make any assumptions, we decided to remove those papers. We run the selection once more, and in this round, the Kappa was 0.89, the strength of agreement almost perfect. Five papers were included and used in the first snowballing iteration.

**Snowballing iteration 2**

In the second snowballing iteration (see Fig. 2.2), 320 papers were collected to analysis and 14 papers were considered as candidates. The first (A1) and the second (A2) authors individually evaluated the 14 papers using the exclusion criteria. For this step, the Kappa coefficient was 0.46, with moderate strength of agreement. One paper was included and used in the next iteration.

**Snowballing iteration 3**

For this iteration, we collected 60 papers, and none of them selected as candidates, concluding the snowballing part of the literature review. Overall, eight additional papers were included from the snowballing iterations, resulting in 32 papers to have their quality assessed before the data extraction begin.

### 2.3.4 Quality assessment

We applied the quality assessment model based on their rigor and relevance scores proposed by Ivarsson and Gorschek [86]. Rigor aspects are related to three rubrics: context, study design, and validity. They were evaluated according to the respective scores, 0.1, and 0.5. Relevance aspects have four rubrics, which are: context, research method, user/subject, and scale. The scores for these
rubrics can be 0 or 1. The paper’s scoring was performed by the first and the second author. To validate this process both authors individually scored the papers and then they discussed the results of each one of them. We performed this discussion due to the possibility of having different perception about the model’s rubrics and consequently the scoring process. After the discussion, the authors agreed on the final score for each paper (see Appendix 2.9.2 and Appendix 2.9.3).

2.3.5 Data Extraction and synthesis

The first author used a pre-defined spreadsheet for data extraction. Table 2.2 shows the data items collected.

We used quantitative analysis and descriptive statistic visualizations to explore and summarize the studies.

To answer the research questions, we applied narrative synthesis [135]. We first classified the practices extracted from the primary studies in codification or personalization strategy, based on their definitions [78]. Second, we identified what KP these practices were enabling. The papers were clear on defining which KP they were aiming to investigate, e.g., knowledge sharing.

Finally, we examined in which organizational layer, described in the proposed framework (see Section 2.5) these practices are performed. Grouping the data into categories helped on exploring patterns and observe how KM practices are mapped in companies that adopted ASD.

2.4 Studies Overview and Descriptive Statistics

Fig. 2.3 depicts an overview of the 32 collected publications, including distribution through the years, publication venues, and type of method applied. We identified primary studies published in 2003 with a substantial increase in the number of publications after 2012 (see Fig. 3c).
We found 14 papers published in Journals, 14 in conferences and 4 in workshops, see Fig. 3a. The venues focus mainly on software engineering, computer science and information systems, except for two journals, Knowledge and Process Management and VINE, that focus on knowledge and information management.

We used the classification proposed by Robson and McCartan [139] to aggregate the studies according to their research methods (Fig. 3b). Case studies represent 34.4% (P1, P2, P3, P9, P10, P12, P18, P19, P26, P30, P31). Five exploratory studies were reported (P11, P20, P21, P22, P28), which had mainly interviews as data collection technique. Four studies report surveys (P13, P17, P23, P25, P32). Grounded theory (P5, P8, P24) and experience paper (P6, P15, P16) had the same amount of studies, representing each of them 9.4% of the studies. We also identified one action research (P4), one experiment (P7) and one longitudinal study (P14). Besides, we found two papers that did not indicate the applied research method (P27 and P29).
Knowledge management strategies and processes in agile software development: A systematic literature review

Overall, we observe that most papers present descriptive studies, which suggests a possible difficulty in conducting studies that require repeated observations along the time, e.g., longitudinal studies. Particularly in KM, real context experiments are difficult to conduct due to the complexity of setting up its environment, time for observation and changes in companies’ routine. This possibly explains why we found only one experiment (P7), and it was conducted with students. Action research also demands time for progressive reflection with individuals within organizations, which aim to formulate solutions to companies’ problems and to test them.

2.4.1 Quality assessment

To classify the papers based on the final score (refer to Section 3.4), we used the categories proposed by Munir et al. [117] to generate a grid of combinations. To score the papers, we applied the scale proposed by Ivarsson and Gorschek [86]. The four categories are:

- A – studies with high relevance and high rigor (1,5 < rigor ≤ 3 / 2 < relevance ≤ 4);
- B1 – studies with high relevance and low rigor (0 ≤ rigor ≤ 1,5 / 2 < relevance ≤ 4);
- B2 – studies with low relevance and high rigor (1,5 < rigor ≤ 3 / 0 ≤ relevance ≤ 2);
- C – studies with low relevance and low rigor (0 ≤ rigor ≤ 1,5 / 0 ≤ relevance ≤ 2)

The relevance aspect relates to the environment where the studies’ results were obtained, more specifically, to the practical environment, industrial context, real applications and applied research method. Rigor concerns the scientific aspects of the reported research.

Fig. 2.4 depicts the distribution of the quality assessment scores. 16 papers were categorized in the B1 area (low rigor and high relevance). It suggests that relevant studies were conducted, but they should have been conducted with higher rigor to increase reliability and validity. One paper appeared in category B2 (P7), with high rigor and low relevance. In category C, three papers present low rigor and low relevance (P27, P4, P15). Category A is the second in the number of papers, with 12 (P8, P14, P19, P20, P21, P22, P23, P24, P25, P26, P28, P31), representing the papers with high rigor and high relevance.
Surprisingly, 59.3% of the studies were classified with low rigor regarding context, study design, and validity evaluation. When authors fail at describing research method and data analysis, reviewers and readers might misinterpret the research [50]. For practitioners, it is hard to analyze the studies’ findings, since they might not be able to compare their context with the ones that describe poorly their context. Therefore, it is not possible to know if the practices adopted to manage knowledge are accurate and valid enough for practitioners’ use. For researchers, the implications relate to how reproducible these studies are and how substantial is the evidence for future work.

Regarding relevance, approximately 87.5% of the studies present high relevance. To this review, this result is significant since we are gathering findings from empirical studies in which industrial settings are more representative. On the other hand, 19 studies present low rigor, what weaken their reliability. They have been unsuccessful in reporting mainly the validity threats. The second rubric that present low scores was study design (see Appendix A), and third, the context description.

Knowledge is a complex and multifaceted theme; possible implications of low rigor on researching this theme could be the misinterpretation for both researchers and practitioners on what is considered knowledge by the companies. This understanding guides not only the research method to be used, but the
companies’ actions to manage individuals’ knowledge. Increasing the studies rigor could contribute on filling this gap and provide stronger evidence that could support future works and more understanding for practitioners.

2.5 Results and Analysis

To aggregate and analyze the results, we created The Knowledge Management Strategy in Agile Software Development (KMS-ASD) conceptual framework to explain how the different concepts are related, see Fig. 2.5.

Framework conception

Software development companies are often affected by changes to the external environment, market and customers’ requirements. As a result, arrangements tend to change from hierarchical to flat, aiming more social interaction within and between organizational layers (strategic, tactical and operational); and changes in how an organization creates value through knowledge, e.g., factory model organization tends to have few knowledge-intensive functions, such as research and development departments [47]. Depending on the arrangements, the distance between the organizational layers may differ. With long distances, the translation of organization goals and requirements becomes a challenge, and may hinder its assimilation across the layers [20]. In the opposite direction, the communication from the lower to the other levels is also hampered.

The arrangements may also be different within the organizational layers, e.g., ASD companies have self-organized and greatly autonomous teams that rely on informal communication. Small to medium organizations, tend to have a flat structure of teams, which facilitates the implementation of KM strategies. Large organizations are often distributed, resulting in a more significant distance between the tactical and operational layers and greater team isolation.

Software development activities do not occur only in the bottom layer of a company, they relate to the other layers, e.g., executives, product management or requirements engineering [58,103,164,166]. In the development process, the value created to the customers is supported by the organizational knowledge [95,136]; which is built by knowledge processes within companies’ routines.

The Knowledge Management Strategy in Agile Software Development (KMS-ASD) framework (refer to fig. 2.5) envisions a high-level perspective of a software development company that has one or more software-intensive products and develops them through ASD processes. Vähäniitty and Rautianen [164] propose a five-level conceptual framework (business, product service, development portfolio, project, and iteration) that aims to link long-term product
and business planning. We adapted their organizational levels by encapsulating business and product service as a Strategic layer; naming development portfolio as a Product portfolio layer; and, product and iteration as a Project layer. This decision was made to simplify the framework and align it with the organizational structure proposed by Mahesh and Suresh [111], that has knowledge as central asset of an organization.

Figure 2.5: KMS - ASD

In the KMS-ASD framework, KM strategies reflect (I) the corporate strategy, which defines how the company should compete for obtaining or maintaining competitive advantage [33,51]. The KM strategies promote (II) at least one of the KPs [2,146] (KC - knowledge creation, KS – knowledge storage/retrieval, KT – knowledge transfer/sharing, and KA – knowledge application). The KPs take place within and between organizational layers. Codification and personalization strategies are usually combined; however, the distribution is not symmetrical, e.g., the mix could be, respectively, 80/20 for codification and personalization [78].

In the Strategic layer, the defined corporate strategy for obtaining competitive advantage is one input to define KM strategies [24]. The corporate strategy is discussed in terms of strategic positioning, the business model, current capabilities and required investments for new technologies and knowledge. The strategic level discussions also include how a company is going to achieve and sustain competitive advantage of market position.
The Product portfolio layer establishes the connection between strategic decisions and project levels. The strategic decision regarding the business, market positioning and the portfolio of product are taken here. The decisions about how to allocate the companies’ resources into developing various products and what competences and knowledge are needed are also taken. In this level, personalization strategies that promote knowledge creation (KC) could be, for example, more aligned to the conception of a new product or product portfolio and their interaction with currently offered products or other services in an ecosystem. The product offering is developed concerning high level functionality and the product delivery strategy is agreed.

In the Project layer, the development iterations are planned according to the decisions made on the middle-level of the company [164]. Here, KM strategies are focused on practices that improve socialization between individuals inside and cross-teams. The less documentation approach to ASD foresee more interaction and, consequently, individuals disseminate more tacit knowledge.

The right balance of KM strategies could help knowledge storing and sharing from individuals throughout the company [141]. KM strategies may also focus on crossing layers. Launching new software-intensive products requires both technical and domain knowledge. The faster the new knowledge is spread and internalized, fewer delays may happen, and quality may be increased [141]. For companies that develop software connected or not to other products, this integration between KM strategies and the different levels could result in a successful competitive strategy.

In the following subsections, we list the KM practices reported in the primary studies regarding the KM strategy they implement (RQ2) and map them into the organizational layers. In each layer, we classify the practices regarding the type of KM strategy they implement, and which KP they promote (RQ1).

### 2.5.1 Strategic layer

Fig. 2.6 depicts the distribution of KM practices (RQ3) that implement both personalization and codification strategies in KMS-ASD.

We found no primary study that took into consideration the strategic layer regarding KM. A possible explanation for the absence of KM practices in this level could be that the performed practices are informal and not directly connected to the corporate strategy or goals. Practices in the strategic layer could assist in discovering new requirements to create new products or improve existing ones, oriented by the organizations’ vision.
Knowledge is the leading resource for software development organizations and the integration among the different layers of the organization is essential. The reason for that is related to how the organizations create value. To Mahesh and Suresh [111], a flexible organization is recognized as an enabler of a free transit of knowledge by creating a social structure that systematically integrates knowledge from different layers.

Agile software development is suitable for this type of organizational structure due to the low level of formality in its development processes. Also, the encouragement of social interaction facilitates promoting the KPs, which might help ASD teams to rapidly adapt to changes not only in features, but in market demand as well. We believe that KM practices could be applied in ASD contexts and help to integrate knowledge from all layers and to support realizing the corporate strategy. For example, if a company aim to be leader in innovation, KM practices that promote knowledge creation might help on developing new products.

![Distribution of KM practices in organizational layers through KMS-ASD](image)

Figure 2.6: Distribution of KM practices in organizational layers through KMS-ASD

However, based on our findings, it seems that agile teams tend to be isolated from the rest of the organizational layers, which means that KM practices could
work well inside and across teams, but not integrated to the rest of the company. In a recent study [6], developers pointed that after agile adoption, they lost the “big picture” of the product because they are more focused on features do be developed. This finding could explain the lack of integration of KM practices to the other layers.

We suggest that future studies investigate how different organizational structures, e.g., flat and factory model, impact the KM practices integration to the different organizational layers of companies that adopt ASD. Moreover, use KMS-ASD framework to explain how strategies and practices are created and integrated to the strategic layer, mainly how they contribute to the organizations’ competitive strategy.

The lack of integration may hamper the process of measuring the efficacy of the practices, due to impossibility of traceability of the practices in the project layer to the competitive strategy.

2.5.2 Product portfolio layer

We identified two KM practices in the product portfolio layer: one implementing personalization strategy, and other implementing codification strategy.

As a personalization strategy, the “marathon of innovation” is a practice reported by Santos et al. (P24) that promotes knowledge creation. The company used its infrastructure to promote interaction among members from different teams and hierarchy levels to work together to stimulate innovation by developing new ideas. Individuals could, as a reward, dedicate time to maintain and grow the projects originated from their ideas.

The new ideas generation is a relevant topic and intrinsically related to KC. Santos et al. (P24) point out the relevance of the customer participation and the role of companies in enabling customer participation. Customers play a key role in ASD since they provide information about requirements and contribute to socializing their needs and experience, which may trigger the KC process to result in new solutions.

Regarding the codification strategy, wiki-based tools are used to transfer knowledge between different departments. This practice was identified in one study (P6), and allows the connection between different business units, keeping knowledge and information updated. The authors also point out that the tool was intensively used by the developers to store the product documentation and knowledge considered valuable by them.

Both practices focus innovation and knowledge sharing between product portfolio and project layer. To establish a connection beyond project barriers,
patterns of communication and knowledge transfer are required. Strode et al. [156] identified in their study that companies use “boundary spanning artefacts” to enable this communication.

This type of artefact could be helpful on propagating knowledge through organizational layers. However, one main challenge remains unaddressed regarding this aspect, which is to know what knowledge the organization needs to share on what level and across the levels.

2.5.3 Project layer

We identified 40 KM practices that aim attention at the project layer. Regarding personalization strategies, the primary studies report practices that promote KC, KT and KA processes. Codification strategy gathers seven practices, they promote KS, KT and KA processes.

Personalization strategies

The KM practices that implement personalization strategies are divided into the following knowledge processes: Seven in knowledge creation; 25 in knowledge transfer/sharing; and one in knowledge application.

Knowledge creation process. Knowledge is created through KM practices that stimulate socialization of tacit knowledge. Three of the reviewed studies (P5, P22, P1) report KM practices for KC. All three studies have high relevance, but studies P1 and P5 have low scores for study design, discussion about validity threats, and context description. It hampers the comparison to other studies’ results and the understanding about on other similar context KC practices could be suitable; for instance, the comparison between P5 and P22 regarding the context where communities of practices are applied.

Bahli and Zeid (P1) argue that the software development process adopted intervene on KC. After conducting an empirical study in industry, the authors concluded that the adoption of extreme programming facilitated KC.

Formal and informal learning practices reported in one primary study are applied to stimulate knowledge creation during project execution (P5). According to Dorairaj et al. (P5), these practices aim to improve team members’ skills by upgrading their technical and management knowledge. Communities of practices (P5, P22), pair programming (P22), and innovation boards (P22) can be used to stimulate social interaction between the team members during software development. The customer interaction is a KM practice reported as a relevant source of KC (P5, P22). To Razzak and Ahmed (P22), daily customer
Knowledge management strategies and processes in agile software development: A systematic literature review

Involvement throughout the development cycles allow team members to discuss issues and create knowledge through continuous feedback.

Knowledge transfer process. 11 primary studies report the application of different communication channels that stimulate social interaction (P2, P5, P12, P13, P16, P18, P20, P22, P24, P26, P29). In distributed teams, this interaction is even more challenging. To minimize the distance and time zone differences, information and communication tools, such as video conferencing, chat rooms and telephones are adopted (P29, P20, P26). Wendling et al. (P29) argue that technology is essential to connect distributed teams. However, it does not replace face-to-face interaction. Moving members between sites for a short time complements this practice. Boden and Avram (P2) found out that giving the opportunity for team members to work together for long periods in a distributed context, facilitates KT between members from different sites.

Collocating teams tend to reduce the demand for information and communication tools. The focus changes to promote a friendly environment that encourages KT. Communities of practices, workshops, consultancy, frequent meetings, and informal gatherings are practices used to share knowledge among individuals (P2, P5, P12, P13, P16, P20, P22, P24, P26, P31). To Karlsen et al. (P12), the interaction during the meetings promotes discussions about different subjects, tasks, solutions, and estimations. This practice allows team members to share knowledge to build solutions together and discuss their progress and goals. In a case study at Ericsson, Smite et al. (P26) found that communities of practices contribute to develop and increase the knowledge network by the frequent communication between teams. Santos et al. (P25) report one practice called brainwriting, a practice similar to brainstorming. In this case, the ideas were written in papers. According to the authors, a brainwriting session focus on reusing knowledge for problem-solving and on innovating by creating new concepts.

The physical working structure is a concern reported in one primary study. Santos et al. (P24) explain that the physical workspace affects KT effectiveness. Walls that separate offices act as barriers for KT. For them, open workspaces that integrate people and furniture that facilitate working together (e.g., tables for pair programming) provide closeness between members.

Companies adapt the software development process and team configuration to achieve team cohesion and better structure to share tacit knowledge (P5, P8, P12, P18). Ramesh et al. (P18) point that the time established for the development cycles depends on the complexity of the functionalities that the team is working. Because of that, the development cycles are short, but not time-boxed. In the studied companies, the length of the sprints was changed,
2.5 Results and Analysis

aiming to achieve more interaction for KT. The authors also point that *keeping the same team during the whole project* is a practice that benefits team cohesion and KT as well.

Besides the adopted development process, the team configuration also undergoes adjustments. Two studies (P5, P12) reported the adoption of *cross-teams* configuration. To Karlsen et al. (P12), tacit knowledge transfer may be facilitated when the team is composed of individuals that assume different roles and have diverse backgrounds. Besides, an increase of motivation is also observed as result of cross-teams’ configuration.

Practices that increase trust between individuals and long-term collaboration have a positive effect on KT in agile teams (P2, P8, P16). To Moe et al. (P16), *conceive and discuss product goals collectively*, mainly when the teams’ members are gathered, induced the members to understand that they need to trust each other and share knowledge to achieve the goals. This practice is also pointed by Ghobadi and Mathiassen (P8) as valuable for knowledge sharing.

The primary studies report practices that aim to leverage individuals’ knowledge through interaction, resulting in technical knowledge sharing. Studies (P5, P22, P24) report that companies promote *formal or informal technical discussion sessions* to discuss technical solutions. In Santos et al. (P24), one company promote *challenge activities* support learning through interaction, aiming to share and improve programming skills.

*Pair programming* is reported as a beneficial practice in both collocated and distributed teams. The informal conversation during the development offers a good opportunity for the team members to share their knowledge (P7). In an empirical study, Ghobadi and Mathiassen (P8) found that when there is a lack of understanding about the business domain, *to engage experienced and motivated people* could help to share knowledge. Another practice that aims to level up knowledge between individual is the *rotation of members*, which is applied in collocated or distributed teams (P5, P12, P20, P22, P24). The rotation occurs by changing the individuals’ roles from one team to another or rotating team member between sites when the company has distributed development. In Santos et al. (P24), most of the studied companies adopt rotation of members when they need to structure new teams.

The relation of customer with the teams is explored in two studies (P5, P8). In Ghobadi and Mathiassen (P8), *leveraging a good relationship with the client* and *organizing training sessions* with the client are practices that provide opportunities to transfer knowledge in both ways. Dorairaj et al. (P5) suggest that having an *on-site customer* during the whole development helps on providing not only feedback but also collaborating throughout the iterations.
The concept of high-quality knowledge sharing is introduced by Ghobadi et al. (P7), and is related to how useful is the knowledge transferred to the software development activities. In an experiment, they explore how coopetitive reward structures influence the high-quality knowledge. The authors claim that on one hand, cooperation leads to better performance; on the other, competition generates more analytical thinking. By combining them, they found that there is a direct influence on the level of high-quality knowledge.

Six of the studies (P8, P20, P22, P24, P25, P26) received high rigor and relevance scores. In this case, the findings regarding the practices utilized for KT may be considered more consistent, providing better support for replication and practical relevance.

Seven studies (P2, P5, P12, P13, P16, P18, and P29) received high relevance but low rigor scores. Validity threats discussion and study design have the lowest scores. The implication of those scores for KT practices is the impossibility of establishing an appropriate comparison between the contexts. This fact may be more critical for distributed environments, since the contexts may differ significantly. Effectively describing the context where KM takes place may help in comparison and understanding about the frequency of social interaction, goals for transferring knowledge, people involved, type of knowledge, and transfer success rate.

Understand how each company perceive the concept of knowledge is crucial for the reproducibility of the KT practices, which is not successfully reported in the primary studies. The lack of rigor could imply in misinterpretation for researchers and subjects about the concepts and practices goals. Particularly, P7 is evaluated with high rigor, but with low relevance, mainly because of using students as subjects in the experiment. Besides, the applications used for the experiment are unrealistic, do not represent real industry context.

Knowledge application process. This process expresses how the knowledge is transformed into competitive advantage of a company. We found only one study that reports one practice that assists KA in the project layer. Dorairaj et al. (P5) report that the sprint review practice in Scrum helps to identify areas that need improvement for the next sprint and helps to create favorable circumstances for new ideas generation.

We could not find practices enabling KS. One possible explanation is that practices to store and retrieval knowledge are highly dependent on explicit knowledge, which is not the objective of the personalization strategy.
2.5 Results and Analysis

Codification strategies

We found seven KM practices that follow codification strategies: three of them are related to KS, three to KT and one to KA. According to Alavi and Leidner [2], KM practices that implement codification strategies aim to store knowledge through documentation or databases. The goals are to share explicit knowledge, to build up experience, and to contribute with new experience and knowledge.

In knowledge transfer process, we found two studies (P4, P8) that report documentation of project experiences. Dingsøyr and Hanssen (P4) propose a lightweight approach to post-mortem reviews. The teams discuss and document good and bad experiences from the projects and involve stakeholder in the review process. According to Ghobadi and Mathiassen (P8), documented experience supports planning future projects by observing inadequate planning performed in the past.

Information and communication technologies are widely adopted by companies studied in the primary studies (P5, P12, P29, P13, P22). Companies keep track of technical details of the software, project documents and meetings to share with the teams. Wiki-based tools are the most used, together with the tools for knowledge transfer, such as JIRA, ScrumWorks, Confluence, Hudson, Yammer, Github, electronic boards and Redmine. One study reports visual prototyping as a practice to share knowledge (P20); however, we could not find any detail about how this practice is conducted.

We found three studies in the knowledge storage process area (P3, P5, P18, P32). Ramesh et al. (P18) report the case of a company that developed a database that their teams use to store different types of information and knowledge, to report issues and get feedback on solutions. The database is also reported by Gervigny and Nagowah (P32) to store lessons learned and good work practices.

KM systems are used to store content from technical presentations, concepts and technical expertise for further consideration by team members. Software details are stored in the code itself (P5).

Chau and Maurer (P3) developed an online tool to store knowledge based on the wiki technology. It is associated with a database capable of storing structured and unstructured information. Besides, the tool aims to support the experience factory approach through structured knowledge and experiences of learning, which is similar to communities of practice.

We found one codification practice in the knowledge application process area. Dorairaj et al. (P5) studied a company that documents knowledge from projects in a wiki-based tool, aiming to reuse or adapts in future projects in
Knowledge management strategies and processes in agile software development: A systematic literature review

similar context. The teams document knowledge related to issues and solutions, adding detailed explanations.

We were not able to find practices that promote the KC process. One possible explanation is the nature of the KC process, that is about externalizing tacit knowledge to combine it with other explicit knowledge. To do that, tacit knowledge requires being externalized to be stored.

Two papers received low rigor and relevance scores (P3, P4). In P4, despite the use of real industry projects, they use students’ subjects; this scenario does not represent a real industry environment. In P3, the practice of using a tool is undermined because of the lack rigor on validating the tool in a real industry context. Therefore, these practices should be considered carefully by practitioners and researchers for two main reasons: lack of context description and industry representation.

2.6 Discussion and Research Gaps

In the primary studies, we observe that companies combine both personalization and codification strategies. 81% of the KM practices implement personalization (promote social interaction between team members) while 19% of the practices implement codification. This means that the practices of codification strategies act as support for the personalization strategies.

Regarding the KPs promoted by the KM strategies, we observe a tendency on the KT practices to focus on tasks and problem solutions by enhancing face-to-face or digital communication. However, the primary studies do not explicitly explain what knowledge is transferred. Using informal communication creates a risk of losing knowledge or not transferring it. The result is that knowledge gets localized inside a few individual’s minds instead of being propagated.

We found only three empirical studies that focus on KC. This indicates that more in-depth studies are required to understand why ASD needs to promote KC and what knowledge they aim to create. KC is not a trivial process to investigate, and probably the reason of the low number of primary studies found is because it requires substantial time from researchers to observe how the knowledge was created, shared and resulted in a new product, process improvement, solutions, among other outcomes.

Curiously, the KA process is the least investigated by the reported studies. The organizational competitiveness may be affected by the ability to effectively apply knowledge [145]. Future studies focusing on ASD could approach where knowledge is applied and how it has become a solution or a product.
2.7 Threats to Validity

In KS, we notice that software companies still struggle to know what knowledge they should store. More important than having a tool, is to know what to store regarding information or knowledge. Additionally, researchers should explore how one can adapt storage and retrieval activities in ASD, more specifically, when these activities occur in the project and what are the guidelines the teams should follow to perform them.

To summarize, it is surprising that the identified papers do not discuss KM practices efficacy. Are these practices effective? A previous study conducted in 2008 also presented this concern [21]; however, our review reveals the same lack of studies regarding the efficacy of KM practices. We recommend to apply empirical and rigorous methods for investigating KM in ASD, such as case studies, followed by in-depth interviews and observational studies since the beginning of a project till it ends, or even longitudinal studies. By doing this, it is possible to trace the knowledge throughout the development process, its externalization, combination, and internalization inside/across the teams/company. Another viable research method is the ethnographic study, also suggested by Bjørnson and Dingsøyr [21].

2.7 Threats to Validity

External validity remains a significant concern since we studied the intersection of ASD and KM. We consider that our findings are generalizable to companies that adopt ASD principles through the different methodologies in software development. However, a risk remains that there could exist studies in ASD or KM literature that could be considered relevant but do not explicitly consider both areas. Due to the systematic nature of this study, we consider this risk to be low.

Concerning internal validity, we address the five characteristics: study selection, subjectivity on quality assessment, misclassification of primary studies, no exclusion of papers with low rigor/relevance, and database selection.

In the study selection phase, disagreements may have led to the exclusion of relevant papers. To mitigate this risk, the two first authors independently applied the exclusion/inclusion criteria and measured the related agreement through the Cohen’s Kappa coefficient [37].

The second characteristic is a degree of subjectivity on quality assessment. Although the model [86] used to evaluate the quality of the primary studies presents rubrics with clear description, understanding them and scoring each paper could introduce misjudgment on quality evaluation. In this case,
the first and the second author performed this process together, discussing the model rubrics and how the primary studies satisfied them.

Referring to the misclassification of primary studies, the lack of proper description about KP and KM strategies described by the authors in some of the investigated primary studies could have jeopardized the reliability of our results. To mitigate that, we defined all theoretical concepts and terminology for both constructs, ASD and KM, and classified the papers during the synthesis base on these definitions.

Regarding the decision of not excluding primary studies with low rigor/relevance, it could have weakened the evidence of our findings. However, excluding these papers, we could miss relevant or rigorous findings, mainly because all of them report empirical studies. Since they already underwent peer-review, we believe that we keep a balance between volume and validity of our findings.

To avoid the risk of missing papers that are not indexed by the selected databases, we complemented the database search with the snowballing procedure. This approach helps reconsidering papers that were excluded by mistake on previous steps, since several of them emerge again during the procedure.

### 2.8 Conclusions and Implications

Comprehending KM concepts becomes more important as companies start to consider knowledge as one of the main resources for building products. Successful KM strategies implementation involves knowing what goals the company aim to achieve, and its relation to the corporate strategy. Thereafter, the KM strategy promote the KPs through practices in the different layers of a company.

By following a systematic procedure, we conclude:

- KM strategies in ASD promote mainly KT process with practices that stimulates social interaction to informally share tacit knowledge;

- The primary studies reported practices in product portfolio and project layers. Most of the practices (95%) are placed in the project layer and the studies do not report about the connection of KM strategies and practices between the strategic layers and product portfolio or project layer;

- 81% of the reported KM practices implement personalization strategy, while 19% of the KM practices implement codification strategy (acting as a support for personalization).
2.8 Conclusions and Implications

- Personalization strategies promote KC by stimulating close interaction between team members and customers. Personalization strategies also promote KT by stimulating social interaction inside and between teams, modifying the agile method adopted to accommodate knowledge transfer practices, and customer participation; and promote KA process by using review sprint approach.

- Codification strategies support storing of the project documentation, learning materials, and meeting notes. KT process is supported by several documents, i.e., post-mortem reviews and prototyping. The codification strategies promote KS using databases and information systems, i.e., wiki; and promote KA by using the knowledge stored in wikis.

The results of this review have implications for both researchers and practitioners. For researchers, we identified several potential research gaps in Section 6. We summarize our indications for future research into three main implications: 1) to investigate how KM strategies are planned in ASD; 2) to explore how organizational structures (layers and size) impact KM strategies implementations in ASD context; and 3) How to define measures for KM practices’ efficacy in ASD.

For researchers, the results could contribute to a better comprehension of how the strategies are related to the processes through KMS-ASD framework, and on which gaps future research should focus. For practitioners, this study offers several insights for KM practices in ASD context, depending on what KP they mainly want to promote. The combination of KM strategies shows that personalization strategies are predominant in ASD. The framework could guide practitioners in planning KM activities that support the corporate strategy at all organizational layers, considering what knowledge should be transferred from and to the different layers.
2.9 Appendix

2.9.1 Selected papers

Selected papers to perform the SLR, as detailed in Section 2.3 of Chapter 2:


Knowledge management strategies and processes in agile software development: A systematic literature review


2.9 Appendix


2.9.2 Rigor evaluation

2.9.3 Relevance evaluation
### Table 2.3: Rigor scores

<table>
<thead>
<tr>
<th>Paper ID</th>
<th>Context (C)</th>
<th>Study Design (S)</th>
<th>Validation (V)</th>
<th>C+S+V</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>P2</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>P3</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>P4</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P5</td>
<td>1.0</td>
<td>0.5</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td>P6</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>P7</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P8</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>2.0</td>
</tr>
<tr>
<td>P9</td>
<td>0.5</td>
<td>1.0</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td>P10</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P11</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P12</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P13</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>P14</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P15</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>P16</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>P17</td>
<td>0.0</td>
<td>0.5</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>P18</td>
<td>1.0</td>
<td>0.5</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td>P19</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P20</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>2.0</td>
</tr>
<tr>
<td>P21</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P22</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P23</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
<td>2.0</td>
</tr>
<tr>
<td>P24</td>
<td>1.0</td>
<td>1.0</td>
<td>0.5</td>
<td>2.5</td>
</tr>
<tr>
<td>P25</td>
<td>0.5</td>
<td>1.0</td>
<td>0.5</td>
<td>2.0</td>
</tr>
<tr>
<td>P26</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>3.0</td>
</tr>
<tr>
<td>P27</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P28</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>2.0</td>
</tr>
<tr>
<td>P29</td>
<td>1.0</td>
<td>0.5</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td>P30</td>
<td>0.5</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P31</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>P32</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>Paper ID</td>
<td>User/ Subject (U)</td>
<td>Context (C)</td>
<td>Scale (S)</td>
<td>Research Method (RM)</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------</td>
<td>-------------</td>
<td>-----------</td>
<td>----------------------</td>
</tr>
<tr>
<td>P1</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P2</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P4</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P5</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P6</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P8</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P9</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P10</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P11</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P12</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P13</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P14</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P15</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P16</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P17</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P18</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P19</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P20</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P21</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P22</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P23</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P24</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P25</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P26</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P27</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>P28</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P29</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P30</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>P31</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>P32</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Knowledge management strategies and processes in agile software development: A systematic literature review
Chapter 3

Thinking Strategically About Knowledge Management in Agile Software Development

This chapter is based on the following paper:


3.1 Introduction

The knowledge of individuals is a well-known competitive resource, and has been philosophically discussed and validated by several researchers within the strategic management field [11, 47, 123, 133]. Knowledge Management (KM) strategies, which are based on knowledge needs and organizational characteristics [24, 141], help with effective knowledge resources management. Successfully
employing the knowledge resource contributes to product and company growth [160].

Software development is significantly dependent on exploiting the knowledge resources [21,141]. However, the Agile Software Development (ASD) empirical literature seems to give more attention to descriptive studies that report the use of tools and practices for knowledge sharing, rather than their effectiveness and impact on the software development [21,127].

We identified in our previous study that the KM practices reported in empirical studies in ASD context have low or no connection with the strategic level of the companies, which has negative implications for traceability between the development practices and the company strategy, and measurement of successful implementation of these strategies [127].

Treating knowledge as a resource implies that it needs to be managed with a logical and structured approach [11,141]. We conjecture that the lack of integration of KM practices with the strategic level, together with the adoption of informal KM practices adds low value to the companies that adopt ASD, and hinders unlocking the full potential that ASD brings.

Based on the recent results of our literature review on empirical studies [127] and on KM theories from the strategic management domain, we discuss future research implications of considering knowledge as a resource in ASD.

### 3.2 Knowledge Management in Strategic Management Domain

The rationale behind using theories is the glue that connect the explanatory factors [168]. Therefore, we base our discussion on concepts originated in underlying theories of the strategic management domain.

The theoretical literature in the referred domain states that the knowledge embedded in organizational routines contributes to company’s effort to achieve its goals, by creating and delivering value to its customers [47,95]. The value created can be product value as well as organizational competence in solving problems and addressing customer needs.

The value creation process has social and cognitive relationships for enhancing individual’s abilities for production. Therefore, organizations arrangements tend to change regarding hierarchical aspects, varying from the factory model to flat organizations. In these different structures, activities are more or less knowledge-intensive, which influence how individuals create value [111].
Several companies perceive the strategic relevance of knowledge as the main competitive resource, especially in knowledge-intensive environments. However, building and maintaining the competitive advantage based on the knowledge resource, requires a strategic plan that integrates the long-term vision with the knowledge resources [75].

Companies are often affected by changes in the external environment, e.g., suppliers, market, laws and customers’ requirements. In this changing environment, companies need to adapt themselves in order to be more responsive and monitor current and future knowledge needs.

In previous work, we envisioned a conceptual framework (refer to Fig.3.1), that displays a high-level perspective of the integration between the corporate strategy and the project level of ASD. The KM strategy reflects the corporate strategy, and promotes, through practices, one or more knowledge processes (KC - knowledge creation, KA - knowledge application, KT - knowledge transfer, KS - knowledge storage) through the company’s levels.

The corporate strategy guides how a company acquires and distributes its resources, e.g., if the company aims for product innovation, which investments in technology and knowledge should be made. In each level, KM strategies might influence different knowledge processes, depending on knowledge needs and their sources [127].

In a study in the manufacturing and services sectors, Ferraresi et al. [63] provided empirical evidence that the business performance is positively impacted
by KM strategies only when these strategies are connected to a strategic orientation. Product quality is also affected by KM. Lee et al. [175] investigated in several industry sectors how KM is linked to product quality. They found that product quality is significantly affected by how process management moderates customer knowledge acquisition and the participation of employees in KM activities.

KM effectiveness seems to have a relation to the management encouragement. The rationale behind a KM implementation involves an internal analysis of the company’s resources and its needs. Then, the decision regarding which tools and methods to use to achieve the selected goals need to be made [26].

The literature gathers three common approaches for conceiving KM strategies: The rational approach - considers an analysis of the companies resources and need, and external environment of the company, such as market and competitors. The emergent approach - developed based on the employees’ daily activities, for example, their methods of problem-solving and their knowledge needs. The integrated approach, which is the combination of both rational and emergent approaches - it is a dynamic interaction where the strategic level provides guidance with the company’s general vision, supported by inputs from the lower level [24].

In any of the three common approaches, we notice that the connection with the corporate strategy of the company is recurrent. Another important aspect that should be considered is the domain where the KM strategies will be executed, by reason of companies size and hierarchy, and knowledge intensiveness of the activities.

3.3 Knowledge Management Strategies in Agile Software Development

Most of the software development activities are knowledge-intensive [52, 143], which evoke the need for the companies to make efforts towards leveraging individuals’ competence throughout the organizational layers. However, when it comes to KM practices in ASD, there is a lack of connection between these practices and the corporate strategy [127].

In a previous literature review on empirical studies [127], we mapped the KM practices in software companies adopting ASD (Refer to Figure 3.2) and which type of strategy these practices follow. Most of KM practices in ASD are focused on personalization strategies, which focus on human interaction to
3.3 Knowledge Management Strategies in Agile Software Development

communicate knowledge, and the majority of them are established in the project layer.

![Diagram of KM practices in organizational layers through KMS-ASD](image)

Figure 3.2: Distribution of KM practices in organizational layers through KMS-ASD [127]

Moreover, there is a tendency of isolation regarding agile teams when it comes to KM, see Figure 3.2. Most of the KM practices might work well inside the teams; however, these practices are informal and occasional. The lack of connection between KM practices and the corporate strategy can result in deviations from the core vision, wasted resources and irrelevant knowledge acquisition that adds no value to the company [11, 25, 141].

The corporate strategy guides the resources allocation for achieving organizational aims [24, 127]. In the resource-based view of the firm, Barney [11] emphasizes that the means of how a company allocates and uses its resources confers the competitive advantage that differentiates the company from its competitors.

Knowledge is an asset that requires comprehensive and logical management to obtain benefits, since, together with skills, it is the main resource of software development organizations [141]. Steen [153] argue that in software development, the product quality phenomenon is heavily dependent on knowledge and skills, however, few research explores this relation.

Santos et al. [145] observed, in an empirical study, that in an ASD context, knowledge sharing effectiveness has relation to purposeful practices, organizational conditions and the stimuli that individuals have to share.
Similarly, goal-modeling may be used in the beginning of software projects to align the system to the organizational goals, which also might indicate that goal-oriented practices are a positive approach to further effectiveness measures for KM; why the practices are needed for; how do they connects to corporate strategy and customer needs.

Both empirical findings and theoretical dialogue connect our discussion, within which the strategic management aspects of having knowledge as a resource, have implications for KM strategy planning.

3.3.1 Potential Research Opportunities

Since 2010, publication related to KM in ASD gained diversity regarding KM focus, e.g., practices, challenges, and theories. Despite that, the state of the research remains far from the KM mainstream in strategic management studies.

We highlight the following research opportunities for exploring KM in ASD:

- **Strategic KM.** Concerns regarding KM effectiveness were raised in previous studies in software engineering [21] [127]. Knowing that KM practices produce the desired results might be crucial to a company on deciding to invest in KM. The two essential aspects to consider in planning KM strategies for a company are the connection between the corporate strategy with the organizational arrangement; and the long-term goals of the KM strategy. [24]. Illustrating these elements in ASD contexts, we could explore: how KM strategies comply with coordination? what adaptations are necessary? Could goal-oriented KM practices facilitate KM effectiveness measurement in ASD?

- **Product quality.** A KM resource remains valuable to the extent that it can deliver value to the customer, and also contributes to achieving enhanced performance [160]. Empirical research has shown that the degree of participation of an employee in activities related to knowledge dissemination impacts the quality of new products significantly [175]. Steen [153] found that software product quality cannot be entirely formalized, but rely on, to a great extent, the practical knowledge, and experience of individuals. Since knowledge is the main resource for software development, future research should keep attention on how to manage the knowledge resource, in ASD context, in a way that it provides superior customer value. What KM activities result in better product quality? In what context? How
these activities affect quality in the software development process, such as requirements, implementation, testing, validation and verification?

3.4 Conclusion

In this vision paper, we discuss the potential research opportunities of KM perspective in ASD. Our inference is that we have reached a degree where the research demands investigations that go beyond mapping the companies actions against the KM theories, to real planned interventions with companies. Knowledge is socially created and translated into processes and products, representing unique characteristics that every company has. Future research should aim to gather more empirical evidence regarding effectiveness, and substantial impacts of KM strategies in coordination and other aspects, such as software quality.
Thinking Strategically About Knowledge Management in Agile Software Development
Chapter 4

Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

This chapter is based on the following paper:


4.1 Introduction

Agile software development (ASD) welcomes responding to change over following a plan. ASD adoption impacts the organizational structures in several ways, e.g., coordination, culture, roles and communication [156].
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

Software-intensive product development companies - for brevity called software companies in this paper - who adopted ASD aim at continuous assimilation of change by sharing, codifying, and transmitting knowledge to people inside and across teams. However, inefficient utilization of the knowledge resource, which is shared through poor codification and informal communication, result in a significant knowledge loss and time waste [148].

ASD puts less effort into traditional coordination mechanisms such as upfront planning and extensive documentation, to gain the necessary agility to better respond to the market changes [156]. It relies on self-organized teams to manage their work mainly through informal communication and the use of tacit knowledge [21] that is shared in an ad hoc manner among people within the teams [68,114].

Although tacit knowledge is recognized as a relevant resource for companies and crucial for innovation [122], it is only possible to be revealed when it is applied [73]. This knowledge is acquired by experience and transferred among people, which is costly, slow and uncertain [96].

Software development is undoubtedly a knowledge-intensive activity [12,70], often associated with complex and intangible social resources that are difficult to reproduce and give heterogeneity to a company within an industry. At the same time, software developers’ knowledge can not be owned, but it should be managed and utilized as one of the key survival factors for software companies [153].

As the software value chain is knowledge-based due to the high dependency on people [141], knowledge application in software production might be jeopardized by the lack of practices to manage knowledge as a resource. Therefore, it becomes essential for a company to know which Knowledge-Based Resources (KBRs) allow the company to have a continuous readiness to quickly respond to the market by intrinsically changing and offering customer value [39,115]. KBRs are tied to a company for a long time. They might appear as specific skills such as technical, creative, and collaborative, whereas the latter regards to integration and coordination of multidisciplinary teamwork [115].

Knowledge in a broad perspective has been explored in the Agile Software Development (ASD) literature [128], and also recognized as a relevant support for managing dependencies in coordination [113]. The studies, however, do not explain how knowledge constitutes a keystone for analysis and further application for companies. Although they recognize the importance of knowledge in software development activities [21,141], software engineering literature lacks studies that evaluate the effectiveness of Knowledge Management (KM) practices or seek to explain the role of the knowledge resource in the ASD [128].
Our work contributes to addressing this gap by understanding how knowledge-based resources support coordination in agile software development contexts. We focused on the planning and the coordination activities utilized by practitioners.

This paper makes the following contributions:

- An empirical investigation to identify the knowledge-based resources in agile software development;
- The development of the Continuous Assimilation Model (CHASM), which illustrates how knowledge-based resources support coordination in ASD;
- A discussion on possible implications for future research and potential solutions for managing the knowledge resources.

This paper is organized as follows: In Section 4.2 we present a brief background and related work. Section 4.3 describes the research method. In Section 4.4 we present CHASM, along with a description of each category. In Section 4.5 we discuss the implications of the findings for KM in ASD. In Section 4.6 we discuss the threats to validity. Finally, in Section 4.7 we present our concluding remarks.

4.2 Background and related work

In this section, we present the basic concepts approached in this research, including an overview on ASD, basic concepts about KBRs, how knowledge relates to ASD, and related work.

4.2.1 Agile software development

Agility and flexibility in responding to changing market and uncertainty [170] are focal points of “Agile Software Development”, which is an umbrella term that includes several methods and frameworks for guiding software development practices [13,39,57,167]. ASD prioritizes human factors such as interaction between people, and teamwork. Further, it focuses on delivering working software when new or updated requirements are expected even late in the development [13].

Dybå and Dingsøyr [57] analyzed the ASD literature and found several contradictory findings regarding the benefits of ASD. Some of the studies report
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

benefits and improvements regarding productivity, while others report the opposite. Benefits from the adoption of ASD were observed in relation to, e.g., collaboration with the customer, estimation, and focus on the work. On the other hand, limitations also were reported, such as not enough attention to architecture and design, and inefficiency of pair programming.

According to Conboy [39], ASD has a significant conceptual gap. To contribute on filling this gap, Conboy developed a taxonomy, based on the organizational and management literature, of agility in information system development. To contribute to agility, an agile method should at least facilitate one of the following: the creation of change, proaction in advance of change, reaction to change, and learning from change. Also, the agile method of choice should contribute to perceived economy, quality, or simplicity without poor performance in any of them.

In a recent study, Annosi et al. [8] observed that the time pressure imposed by the Scrum framework had negative implications for learning, and innovation. People keep focused on the tasks to develop at the cost of losing the big picture of the product. Li et al. [108] also discussed the negative effects of time pressure.

Dealing with constant changes is the central idea of ASD, and knowledge plays an essential role in this regard[141]. While ASD prioritizes informal communication due to the flexibility of the processes and reduced documentation, it relies to a large extent on knowledge that is shared among people [36].

Previous literature points out that companies who adopt ASD utilize mainly tacit knowledge [21], and rely on informal communication to share this knowledge between people in both co-located and distributed ASD [49,69,114]. Although tacit knowledge is recognized as a relevant resource for companies and crucial for innovation [122], it is only possible to be revealed when it is applied [73]. This knowledge is acquired through practices and transferred between people, which is costly, slow and uncertain [96].

However, in a industry where knowledge is the main competitive resource, the lack of management practices might jeopardize the application of knowledge in the production of goods and service. In the interest of advancing the research about knowledge-based resources in the software industry, our study expands the understanding about what the knowledge-based resources are and how they support the ASD.

4.2.2 Knowledge-based resources

A company’s competitive strategy, according to the resource based view of the firm [12], is substantially dependent on how the management maximizes the
produced value, by allocating its resources and capabilities. These resources are idiosyncratic and provide heterogeneity to a company in a market. The knowledge resource gets higher attention when it is involved in activities that require human interaction for production, inside and outside the companies, e.g., teams, and customers collaboration [9][149].

The knowledge-based view of the firm [73] derives from the resource-based view, and focuses on the people’s knowledge as the most strategical resource of a company. Knowledge-based resources might appear as specific skills such as technical, creative, and collaborative, which, in turn, regards to integration and coordination of multidisciplinary teamwork [115].

KBRs have usually been researched in strategic management studies. Most of the empirical research in this field focuses on the industry domain level; while research in lower levels, e.g., company unit, and departments, are rare. Regarding the conceptual studies, they attempt to generalize, even though the resources might have different values depending on the considered context [115].

4.2.3 Definition of relevant concepts

In this subsection we describe the concepts utilized in this study, and the relevant premises that further function as foundation for our developed theory.

The first concept is knowledge. The nature and different understandings about the concept of knowledge are extensively discussed among philosophers throughout the literature [73]. However, this discussion is not our main purpose. We consider a straightforward concept of knowledge as “a collection of information that provides guidance” based on people’ cognitive processes.

The types of knowledge are also largely debated, although, this study considers the two most used types: explicit and tacit. Explicit knowledge is systematized in formal language (e.g., manual and specifications), and tacit knowledge is highly personal rooted in actions values or routines, which is difficult to formalize and systematize in formal language [124].

In its abstract and tacit form, knowledge is highly challenging to manage. Due to this condition, organizations use KM as tool to extract knowledge from people so they can have ownership of this resource. We use the definition that states that knowledge management is the organizations’ efforts to manage the workforce’s knowledge through social processes, aimed at extracting tacit knowledge and transforming it into valuable resources, and the application of tools and techniques to manage the systematized knowledge [12, 73, 83].

The KM is supported by four knowledge processes: creation - combine tacit and explicit knowledge for new knowledge generation; codification/storage -
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

means for systematizing and storing relevant knowledge as useful information; transfer/sharing - move both types of knowledge between people or groups; and, application - which involves the use of knowledge to generate value [2,42].

4.2.4 Related work in KBRs

In this subsection we briefly describe the studies focusing on KBRs. To the authors knowledge, there are no studies discussing KBRs in the software engineering field. Most of them are placed in the strategic management field.

A study with Hollywood film studios from 1936 to 1965 has shown that the knowledge-based resources contributed to return on sale, operating profit, and market share [115], while this industry was in a period of uncertainty. In the same study, the authors point that the value of the knowledge-based resource for the related industry is context dependent, and that varies with the macroeconomic environment.

In the hotel industry, the knowledge-based resources that are more relevant and enhance different types of innovation are related to coordination of activities, external partnerships, business processes and methods, and employees’ skills. They contribute to innovation regarding service delivery methods, product innovation, and organizational innovation [121].

Kaya and Patton [92] explored, in different industry domains, the relationship between knowledge-based resources and innovation performance. They analyzed several knowledge-based resources, e.g., technical expertise of the staff regarding management, customer service, marketing, new ideas generation, product development, and staff commitment to the company. The results revealed that knowledge-based resources significantly enhance innovation performance.

In a study about the determinants of strategy and performance of the British export market [16], the authors found that knowledge-based resources such as international market operations, international market, knowledge about customers and competitors are significant sources of variation in a company’s performance. Knowledge about resource management reinforce the company ability to spread its operations domestically and to foreign markets.

All company resources are relevant, although, they contribute in different levels to the performance and competitiveness [12,16]. To maximize these contributions, companies must know how valuable these resources are, why they are valuable, and in which circumstances [4].

Knowledge has been considered an essential resource in software development [21,141]. Miller and Shamsie suggested in 1996 that research should focus on clarifying the impact of knowledge-based resources in turbulent environments.
such as the software industry [115]. However, to the extent of our knowledge, no research has explored knowledge-based resources in ASD.

4.3 Research method

The goal of this study is to understand how knowledge-based resources support coordination in an ASD context. We are particularly interested in the agile practices and its mechanisms applied to coordinate activities.

The formulation of the Research Questions (RQ) were guided by the findings and identified research gaps of our previous systematic literature review [128] and initial discussion and brainstorming with three companies):

- RQ1: What are the knowledge-based resources in ASD?
- RQ2: How do these knowledge-based resources support coordination in ASD?
- RQ3: How are activities connected to and/or dependent on KBRs in ASD?

To answer these questions, we investigated the planning and coordinating activities utilized by practitioners in software development companies that adopt ASD. We focused on these activities because the coordination mechanisms stimulate the emergence of conditions where people are able to combine their specialized knowledge [73].

We chose to conduct a qualitative study to efficiently capture data regarding the complex phenomena of human behavior [85, 147].

We followed the principles of constant comparison method for theory generation - Grounded theory [40, 41, 147]. As both epistemological and ontological positions in this research, we adhere to the social constructionism, which considers social phenomena to be constantly changed and affected by human interaction [28]. Additionally, we comply with the view that recognizes knowledge as being socially constructed [122].

We collected the data over interviews and documents. We examined the data through open coding (text and audio) to create the concepts that are part of and explain the categories established [147]. The method detailed by Corbin and Strauss [40, 41] allowed us to identify the knowledge-based resources and understand the phenomena behind the use of these resources in the software development.

Figure 4.1 depicts an overview of the research process. Each phase is identified with the subsequent described subsections. The research process was
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

iterative, each piece of data collected was immediately analyzed and provided more questions to be explored in the successive data collection. We reached theoretical saturation, which means, no more new concepts were found, and the data analysis advanced to the creation of the explanatory categories until we reach the model.

To report the findings, we followed the guidelines proposed by Stol et al. [155], which provide guidance to report grounded theory in software engineering. The additional documents such as interview guide, and informed consent, are respectively in the supplemental material.

Figure 4.1: Research Process overview

4.3.1 Research design

The first phase (see Fig.4.1) comprises the initial discussions about the research design, research questions, and method selection. The idea was proposed by the first author and refined through iterations with the other three authors during February and March of 2018.

The subsequent subsections describe the sampling of practitioners, method for data collection and execution, data analysis, and the ethical aspects that we considered when planning and executing this research.

Sampling of practitioners

Practitioner selection was performed by convenience sampling [101], focusing on participants that performed activities connected to coordination. To ensure that the different parts of the software process were covered, we sampled practitioners with roles and responsibilities in the different phases of the software...
4.3 Research method

Table 4.1: Practitioners description

<table>
<thead>
<tr>
<th>Participant</th>
<th>Role</th>
<th>Education background</th>
<th>Years of experience</th>
<th>Company Domain</th>
<th>Agile method</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Architect</td>
<td>Computer Science</td>
<td>7</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P2</td>
<td>Program Manager</td>
<td>Computer Science</td>
<td>17</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P3</td>
<td>Design Leader</td>
<td>Computer Science</td>
<td>8</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P4</td>
<td>System Manager</td>
<td>Software Engineering</td>
<td>18</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P5</td>
<td>Product Owner</td>
<td>Software Engineering</td>
<td>3</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P6</td>
<td>Scrum Master</td>
<td>Computer Science</td>
<td>5</td>
<td>Management applications</td>
<td>SCRUM</td>
</tr>
<tr>
<td>P7</td>
<td>Software Engineer</td>
<td>Development and Analysis of systems</td>
<td>3</td>
<td>Consumer electronics</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P8</td>
<td>Lead Architect</td>
<td>Computer Science</td>
<td>21</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P9</td>
<td>Product Manager</td>
<td>Engineering</td>
<td>12</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P10</td>
<td>Developer</td>
<td>Computer Science</td>
<td>27</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P11</td>
<td>Line Manager</td>
<td>Computer Science</td>
<td>17</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P12</td>
<td>Platform Maintainer</td>
<td>Computer Science</td>
<td>20</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P13</td>
<td>Senior Developer</td>
<td>Software Engineering</td>
<td>15</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P14</td>
<td>Senior Quality Engineer</td>
<td>Software Engineering</td>
<td>7</td>
<td>Consumer electronics</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P15</td>
<td>Scrum Master</td>
<td>Software Engineering</td>
<td>12</td>
<td>Telecommunication</td>
<td>SCRUM - Partially</td>
</tr>
<tr>
<td>P16</td>
<td>Team Lead</td>
<td>Engineering</td>
<td>15</td>
<td>Mobile communication</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P17</td>
<td>Manager</td>
<td>Acoustics Engineering</td>
<td>15</td>
<td>Mobile communication</td>
<td>Kanban /SCRUM</td>
</tr>
<tr>
<td>P18</td>
<td>Lead Engineer</td>
<td>Computer Applications</td>
<td>9</td>
<td>Mobile communication</td>
<td>Kanban /SCRUM</td>
</tr>
</tbody>
</table>

development, as well as in different organizational levels. We intentionally sampled practitioners in companies that work with both co-located and distributed development, companies that have software development as their primary business, and companies that have software development as an activity complementary to the main business. In total, we have collected data from 18 practitioners from five companies (see Table 4.1).

The adoption of this sampling strategy allowed us to have a broad perspective about the knowledge-based resources in the different contexts where the agile practices are applied. The practitioners are from companies located in Sweden and Brazil. The domain of these companies are: telecommunication, consumer electronics, mobile communication, and management applications. With the exception of one company that does business only in the Brazilian territory, the rest of the companies have worldwide activities.

Data collection

Since knowledge is embedded in daily routines of practitioners in companies, the more comprehensive understanding about this phenomenon was facilitated by using multiple sources of data. We also chose this approach to comply to the research method chosen and to be able to triangulate the sources of data [41,161].

The data collection was made through three different sources: interviews, questionnaire, and documents (e.g., quality framework and coordination scheme). The first author created the initial version of the interview guide and similar to phase 0, there were discussions and iterations to formulate the questions aiming
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

at gathering information about the knowledge resource behind, for example, activities such as breaking down the tasks.

The interview guide and the questionnaire were adjusted according to the role of each practitioner interviewed, e.g., product owner, developer, scrum master, line manager, project manager (see supplemental material A).

To collect additional information from practitioners that were already interviewed, we sent a follow-up questionnaire through e-mail. These follow-up questionnaires were adapted based on which information we needed.

The interviews lasted between 30 and 60 minutes, 16 were on-site, and one through Skype®. One questionnaire was sent through e-mail. For the questionnaire we agreed with the practitioner on one week for answering it and returning to us.

The data collection started in May of 2018. We collected it in batches. The first five interviews were performed and analyzed in May - Jun. In this first batch we checked the changes that we needed to do in the interview guide and in the subsequent ones we evaluated what we could explore more substantially on the forthcoming batches.

In July (second batch), the first author performed one interview through Skype® and sent/received one questionnaire through e-mail. In August, the first author conducted seven interviews (third batch). The data collection lasted till November of 2018, when we performed the fourth batch with four interviews.

Ethical concerns

Although our study did not process sensitive data detailed in the Swedish Personal Data Act (1998:204) that would require an ethic board review, we followed the ethical considerations specified by the Swedish Research Council [158].

Regarding the data collection, we paid attention to confidentiality by limiting the access of the data to the authors, and non-disclosure of participants’ name, gender or nationality. In the interview guide, we were cautious about questions that could be emotionally intense or cause psychological harm to the participants [3].

Before each interview or questionnaire, the first author explained the purpose of the research, which is stated in the informed consent (see in supplemental material B). We asked permission for audio recording during the interviews, and explained who would have access to the data, how we were going to use it, and for how long we would keep the audio recording.

When sampling participants, we focused on the roles that were relevant for our research in terms of activities that the participants perform and how
these activities were connected to coordination. The companies agreed on participating due to their interest on the results of the research. Therefore, we avoided economic, legal or power structure dependency between the authors and the participating companies that could severely influence the participants’ responses. Based on these criteria, the companies sampled the participants that were available in the period that we visited them.

Our ethical concerns regarding the data analysis include the stigmatization or harm to specific populations. In this regard, we did not collect or analyzed data considering gender, race or minorities.

4.3.2 Data analysis phase

The first author conducted the coding procedure in each round of analysis, which was carried out after each batch of data collection. The results were discussed with the second author during the entire process of data collection and analysis. As an example, the first five interviews were analyzed and we already observed a few phenomena happening over and over, indicating a future category. On the subsequent batch, we observed new concepts appearing, but also reoccurring ones when comparing to the first batch.

New concepts and categories appeared as we moved the interview to different roles and industries contexts, and previous ones strengthened. The categories were refined several times and the concepts adjusted after each batch of data collection and analysis. Dividing the data collection in batches gave us maturity to reflect upon the data to develop solid results, and also supported us on theoretical sampling [40] by choosing participant by roles that could provide more clarification about the concepts that we generated.

Data storage

We stored all types of data (audio and text) in the software for qualitative and mixed methods research - MAXQDA\(^1\). The software also allowed us to extract the concepts directly from the data source and structure to the categories generated.

Open coding

In this first stage of the data analysis, we observed events, actions and interactions in the coordination activities analytically [41]. We gave them conceptual

---

\(^1\) Available at https://www.maxqda.com
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

labels, which after refinement became the KBRs, that were further grouped into categories. For example, when creating the concept codification structure of documents, we observed reoccurring events when the way of structuring knowledge impacted the reuse of code, association between documents, and the logic behind the description of parts of a system.

Axial coding

In this second stage, we gathered the KBRs that were related to the same phenomenon. Using one of our concepts as example codification structure of documents, we added it to the category Ability to codify and transmit relevant knowledge together with other concepts that helped to determine the conditions where this category raised, in which contexts it was achieved, and what actions/interactions through it occurred.

Five major categories emerged from the data: ability to codify and transmit relevant knowledge, social activities and collaboration, team environment/settings, analytic perspective, and systemic reasoning.

Selective coding

After the third batch of data collection, we started to draw our core category by connecting it to the categories that we created through the ongoing analysis. In our research, the core category did not emerge from the existing categories, which might happen [41]. In this case, we needed a greater abstract concept to explain the main phenomenon and bring the categories together in a model [65], which was the continuous assimilation of change.

We diagrammed the categories and their relationships considering the paradigm features for model generation to explain a phenomenon which considers the following: the conditions where the phenomenon emerges; the context in which it can be projected; the actions/interactions. The core category emerged through a gradual and steady process of reflection. In this process, we discussed the connections among the categories, and we also did several adjustments to a better integration of the model.

4.3.3 Theory evaluation

The criteria for evaluating this qualitative research method were established by Corbin and Strauss [41]. They are divided in two criteria classifications. One for research process criterion and another for empirical grounding findings.
We addressed the criteria by detailing each phase of the research design, and the data analysis. To supplement the validation of the empirical grounding findings, we mapped the model categories to the existing literature. This is detailed through a synthesis in each created category.

4.4 Findings: The Continuous Assimilation Model

We present and discuss the findings in two main chapters. In this chapter, we describe the model that we created, and each category in detail. In section 4.5, we discuss the implications of the model.

We gather the results in the Continuous Assimilation Model (CHASM), which represents the (KBRs) supporting coordination in agile software development environments. CHASM comprises 44 KBRs ($RQ1$) that are classified in six categories:

- Business analytic perspective
- Product systemic reasoning
- Social collaboration
- Ability to codify and transmit knowledge
- Team environment/settings
- Inefficient utilization of the knowledge resource

4.4.1 Model generation

The CHASM model is shown in Fig 4.2, which displays the relationships between the categories. They are represented by identifiers (e.g., C1) and detailed in the next subsections. The model generation is established considering the explanatory elements of the phenomenon specified by Corbin and Strauss [41]:

- The *Phenomenon* is expressed by the core category Continuous Assimilation of Change (C1).
- *Context* is limited to software development companies that adopt entirely or partially agile practices for coordinating the development.
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

The causal conditions that explain the circumstances within which the continuous assimilation of change is supported is represented by the categories: ability to codify and transmit knowledge (C4), business analytic perspective (C2), and product systemic reasoning (C3).

The Actions/interactions by which the phenomenon is expressed are through the categories social collaboration (C5) and team environment/settings (C6).

The consequence of inefficient utilization of the knowledge resource (C7) is knowledge loss and waste.

In a high level view of the phenomenon, our model summarizes how knowledge-based resources support coordination in ASD context:

Software development companies, adopting entirely or partially agile methods, utilize their KBRs aiming at continuous assimilation of change (RQ2) by sharing, codifying, and transmitting knowledge to people inside and across teams. By assimilation we consider the fully understanding of the changes and its implications, and its absorption in the routines.

This phenomenon is supported by people's business analytic perspective (C2), ability to codify and transmit knowledge (C4), and product systemic reasoning (C3). Utilizing social collaboration (C5) and team environment/settings (C6), the companies manage to utilize certain a level of its knowledge resources.
through coordination. However, the inefficient utilization of the knowledge resource (C7), which is shared through poor codification and informal communication results in a significant knowledge resource loss.

Each category has the KBRs highlighted in italic in the following subsections, except for subsection 4.4.7 in which we highlight in italic the consequences of the inefficient utilization of the knowledge resource.

The KBRs might have different levels of importance. As we discussed early, even though two companies have the same resources (both physical and intangible), the heterogeneity between them comes from the intangible resources and how they are employed [12].

Therefore, coordination activities are dependent on KBRs to the extent of how critical these resources are (RQ3), which will vary based on the potential impact of the KBRs on the coordination efficiency.

4.4.2 Business analytic perspective - C2

The business analytic perspective refers to people’s abilities to logically examine external scenarios that might affect the current business models or highlight new business opportunities. It supports continuous assimilation of change (C1) by expanding the analysis of diverse situations, which may include the reaction to market changes by knowing when and how to adapt agile practices according to market conditions in which the company competes, and/or analysis of the impact of the upcoming requirements.

In our study, we observe that the competitive market conditions affect which agile practice companies adopt and/or customize. Competing companies need to speed up their development processes and offer faster release cycles. This shortens deadlines in response to customers’ demands for faster delivery. One subject (P2), who was program manager in a company, stated that “the customer is starting to use it and starting to give us requirements, so these customers are putting heavy pressure on us. In the early days, it was easier, we had more freedom to develop at our own pace. Now the customer puts strain also in the agile way of working because as soon as you have a customer that signs a contract, he expect somethings to a certain date, and therefore, as soon you do that kind of agreement, you kind of destroy the whole agile flow’.

Companies that are leaders in terms of innovative products or operate in less competitive markets, have longer release cycles and focus more on product steadiness and long-term features. In this sense, the agile way of working is customized to cope with more flexibility to innovation than with pressure to deliver.
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

The changes, in general, are mostly related to the continuous addition of new requirements. However, their selection requires a combination of technical capability with market vision to be able to contribute to predicting the future of the products, as remarked by P9: “I came from the client side. I used to program in one of these systems and I think that knowledge helped me. I know how to view the product from the outside and how they want to utilize the product. What trends do we see?”

In addition, since the companies continuously implement new requirements, knowing how the current technology should evolve to support changes becomes one priority, as stated by P8: “That’s one of our roles, to try to see where we are supposed to be in a few years on the technical support level.’

To achieve a sufficient rationale for decisions, practitioners need to balance between business and technical skills. In this regard, P2 comments: “I managed two programs till now, so I am quite new in the management kind of area, and it has been quite interesting for me, to think about how my technical skills help me in this role.” They apply those skills to evaluate the business value in the short term versus long term, but also consider customer value for prioritizing both requirements, and issues and bug fixes.

When the new requirements are incorporated into the product, the comprehension of the implications of changes during the software development becomes necessary in order to verify how these new requirements impact development, testing and deployment time as well, as P1 remarked: “There are technical aspects for features that we introduce that I sort of join to study what would be change to the product, what would be adding to the product, how would impact the deployment and of course the tests. It is quite a lot, actually.’

This comprehension also influences the readiness to absorb changes as pointed out by P5: “The collaborative work also applies to the layers, for example, managers. They all work with the same backlog and need to work together, and make decision on what they need to focus on, which committing to a sprint plan would not work really well. They need flexibility to diverge on the path they take, due to maintenance and testing. They need to focus and fix before continuing towards the goal.’

SYNTHESIS: Some research emphasizes the different aspects of understanding and describing product contextual situation to support software development activities [94,110]. However, how each change impacts ASD practices, and how knowledge resources might support dealing with upcoming requirements, have not been studied. Therefore, we suggest that this set of aspects needs to be enriched by additional elements.
Between organizational/social and technical challenges that software companies face, the former represents the more critical issues such as coordination and communication. Those issues often relate to the lack of skills, that when combined with high rates of staff turnover negatively impacts the organizational knowledge accumulation and competence increment [45, 77, 91].

The concepts described in this section clarify how knowledge based resources can be helpful in minimizing the complications that changes may bring by analyzing their impact on ASD practices. They also highlight the aspects related to knowledge that can assist requirements engineering activities. As an example, instead of extensive focus on requirements elicitation, market-driven companies [72] could associate them with people’s skills that combine technical capability with market vision to maximize the benefit of the technique.

4.4.3 Product systemic reasoning - C3

This category refers to the employees’ understanding of the product as a whole, and the ramification of their actions throughout the product and its associated business model. It supports continuous assimilation of change by providing better coordination when considering the complexity of the product and the ecosystem that this product is deployed in. This category also supports C2 (business analytic perspective) in a sense that can assist the recognition of the internal and external changes that relate to different parts of the system at the same time, for example new laws.

The product awareness is a strong knowledge-based resource that allow teams to reduce waste such as unnecessary code produced as well as time spent on fixing problems that were introduced by the lack of this knowledge.

Missing the product awareness is a phenomenon faced by several practitioners, as pointed by P1: “In my role, as architect, I try to take that value, that part very high, and actually raise those concerns whenever I see them, I try to keep track of all the trouble reports coming, to see if I can find any pattern, how we solve different things, and also of course, try to spread that view to others. Because it is very easy when you have a quite large organization, 70 people is quite a lot. Try to get everyone to understand that, I might be solving my problem, but I might be causing bigger problems.”

P6 also mentioned: “In his project, it is common when focusing on specific features, the developers lose the idea of the product and end up developing unnecessary code that will cost more time on refactoring. People do not stop to analyze what is being done.”
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

The companies establish strategies to handle task planning considering the dependencies to increase the systemic reasoning and awareness. We observed that companies made changes to the agile practices used when several dependencies started to appear, as pointed by P13: “when the project was mainly concerning us, it worked quite well. But as soon we got dependencies and project managers had different priorities, it didn’t work quite well. That’s one of the reasons we switched to Kanban.”

A different strategy to deal with dependencies is to plan the tasks through iterations and prioritize tasks in each iteration. In this regard, P2 comment “there maybe five departments and all of them need to do a piece of work in order for this to be complete. And therefore, this department needs to go first than this one, these two can go on parallel, and then I can do my work. So, there is a lot of planning to this to be done in that way, dependency planning. So, this goes through several iterations, I can say. That’s how we plan our work.”

Product systemic reasoning is reinforced by cumulative experience. People with several years of experience in the company support coordination by possessing broad knowledge about the product. Experienced engineers take decision-intensive roles, e.g., lead architecture, project management and software quality. They are also mixed when forming teams aiming to disseminate the company’s culture of sharing knowledge, routine processes, and mainly knowledge about the product.

Establishing a team requires expertise from managers on how to appropriately distribute human resources to allocate the people to fit the teams’ purpose considering their knowledge about the product. About human resources, P2 comments: “It is actually a combination of all the departments, therefore, you need some teams that we put together that have connections with all other departments, and have forced departments to act, because sometimes these departments might think this is not our problem, this is their problem. Our performance is fine, yes, but if you put them together, the performance is poor.”

SYNTHESIS: Not understanding the big picture of the product and its related business model is a common issue faced by ASD due to the extensive focus on specifically delivering the features in the current sprint and the ubiquitous time pressure [8,15,19,29,89]. It gets more complicated when companies start scaling agile and distributing the development in different locations.

However, scaling agile is a current phenomenon, and dealing with their implications is the most important inquiry [18,120,129]. Product systemic reasoning needs to be managed in both local and distributed development organizations for generating business value and minimizing coordination challenges.
The creation of KM practices is context dependent. For example, transmitting knowledge regarding *product awareness* might be done in different ways; formal visualization techniques in Information and Communication Technologies (ICTs) could be more effective in distributed teams due to the probability of reaching a higher number of people, while on the other hand, informal interpretations of developers in co-located teams could cost less but also could be effective [132].

When the KM practices focus mostly on social interaction, experienced people could help spreading knowledge about the product through encouraging inter-team communication. They can act as knowledge brokers affecting the group’s performance [112]. Balancing social interaction and codification to transmit product knowledge is also important for an organization with high rate of personal turnover.

### 4.4.4 Ability to codify and transmit knowledge - C4

This category relates to people’s ability to recognize what knowledge (tacit and/or explicit) should be codified into artifacts and how. It also includes the ability to integrate coordination skills with tools. It supports continuous assimilation of change by allowing the teams and stakeholders to access relevant and structured knowledge when they need it, without losing much time in the search process.

The different roles on different organizational levels recognize that *knowing what knowledge to codify* is highly relevant but challenging in agile contexts, where informal communication dominates. For example, respondent P3 commented: “There is this thinking, not only in this company, that I write my code and I did my job. But in a large organization it is difficult to maintain a product without documentation. One team develops the code, another team tests it. Then a different team from the first one needs to investigate two or three times more time to start to fix the problem, because they need to first understand what the first team has been done, due to the bad documentation.’

Knowledge is shared, created and applied to a large extent in daily routines. The flexibility in the communication together with collaborative environment promotes the *experience transfer* by combining people with respect to years of experience and background. Quoting P11 “…in fixing the teams, I am looking at ages, experienced people. I try to hire new people to learn from the older.’ The experience transfer promote learning and also knowledge creation by combining different expertise and experience.
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

However, people frequently struggle with knowing when relevant knowledge needs codification for two main reasons: First, identifying what produced knowledge is valuable to others, and second, balancing time allocation for codifying the knowledge and the time pressure for delivery. Regarding this issue, respondent P1 commented: “We really haven’t that culture at all, we have tried to document stuff, but we are so decentralized that if we introduce something it has to cause less friction, and if there is no immediate benefit on it, it won’t fly.”

To ensure that people inside and across teams do not waste time looking for knowledge already created within the company, matching knowledge needs to the codified knowledge should be established. However, we noticed that this is still an open issue in the companies together with the challenge of spreading the awareness of the existing knowledge.

Further, the codified knowledge produced requires update so it can be a living document, as marked by P3: “people have this idea that you do the documentation and that’s it. It is a living document”. Nevertheless, the codification process involves representing knowledge efficiently into an artifact, that is facilitated by a codification structure, so it can be reused and represented in an uncomplicated way, as remarked by P14: “Everything is relevant in some way, but it has to be in a structured way.”

Although there are several tools for managing software development knowledge and the accomplishment of tasks, the ability to integrate tool and coordination skills works as backdrop for breaking down backlog items and disposing them in the tools, as confirmed by P1: “The product owner, he is very detailed, very structured, so it is very easy to know what they are doing, they have a very good planning in the ‘project management tool’, it is very visible, and they are also very good on understanding the big picture and the value they would be adding by solving this.”

In addition, this ability is also associated to codification of architectural knowledge. Due to a high number of activities, architects rely on the design which might be misinterpreted by the team and cause mistakes, as pointed by P18: “it is quite easy to misinterpret the architecture in software development. The architects define the architecture, and they rely on the team to do what is designed.”

SYNTHESIS: Storing information and knowledge in artifacts is a common practice in software companies [34,55,90,98]. Although several ICTs are utilized by agile teams for this purpose, how knowledge resources are effectively employed in the codification processes is not clear in the literature.
4.4 Findings: The Continuous Assimilation Model

The KBRs are mostly represented by skills related to understanding what to store, and also to what extent that knowledge should be codified: entirely physical, digital or mix of both [32, 50, 152]. Software companies who seek to scale agile can not afford to have ad hoc modes of social interaction for sharing the total amount of information and knowledge that the development activities rely on. They need to find a balance between socialization modes and codification processes [46, 59].

Some of the issues emerging from these findings relate to the complexity of increased dependency between teams that are geographically distributed. Distributed environment required more knowledge classification and structures. The concepts described in this category could support coordination in this changing environment by analyzing who would benefit from the bug solution, and what is the best way to transfer this knowledge and in what structure it should be codified?

We expect that incorporating these concepts in the current codification practices, knowledge loss could be minimized by capturing and storing relevant knowledge for further reuse [106].

4.4.5 Social collaboration - C5

This category refers to people’s social skills to engage in information and knowledge exchange and networking. It supports continuous assimilation of change by providing sociable and trustful environments where people experience effective guidance for working routines.

Socialization processes play an essential role within the teams in shaping people’s behavior towards communication improvements. Social collaboration is not about converting introvert people, rather it is about how to manage social processes where sharing becomes priority taking into consideration individual characteristics.

Companies adopting ASD benefit from flat communication to enhance a collaborative culture routine. At the same time, the hierarchical structure regarding departments and roles do not undermine the potential for communication among people, as stated by P14: “you can talk to managers and managers of managers like normal person discussing things, you can discuss with people on other levels.’

The flexibility of communication inside and across teams triggers cognitive processes for combining knowledge from different people and roles to handle the implications of change. In this regard, P14 commented: “When discussing something, maybe two people that do not know the problem, together when
they discuss they solve the problem." In addition, the awareness of the cognitive process is combined with the level of control in activities that involves knowledge creation, which in higher levels hinder the creativity, as noted by P16: "It will stop creativity if you are too formal, it should be a bit loose, but not too loose because it will not gain anything good."

Even though there are diverging opinions on the utility/usefulness of meetings in the agile community, the ability to conduct them might drive people to achieve particular goals when they are established, e.g., combining different perspectives to take decisions. In this regard P2 commented: "My personal hate is reoccurring meetings. If I had a choice, I would cancel all reoccurring meetings, and forbid them. Because 90% are waste of time and could be handled in a better way through communication or whatever. Of course, there are meetings that when issues arise, and it is simply the best way to put everybody in the meeting room, bash their heads together and the issue is solved."

Moreover, when systems from different companies need to integrate (partners or customer), collaboration for coordination of technical knowledge transfer goes beyond breaking down the work to be done. In this scenario, the collaboration aims at solving issues together through cognitive processes that stimulate knowledge creation and its application.

SYNTHESIS: A large part of software development relies upon tacit knowledge [21,127,141,143]. Good social relations are identified as a primary channel to share tacit knowledge and also predict effectiveness in software teams [142].

Team performance is also strongly related to social processes, which are claimed to be major contributors compared to processes methods and technology [62,74].

In this regard, coordination should seek to utilize the knowledge resources described in the category, with the purpose of maximizing collaboration between people. Furthermore, codification techniques (see Section 4.5.1) may also be applied to capture relevant knowledge to spread throughout the company.

Complementary to collaboration stimuli, physical environment has been claimed to improve communication between team members in ASD, specially open workspace [116,134]. However, in a recent study [17], Berntein and Turban refute this claim. Contrary to intuition, they found through a quasi-experiment in industry that face-to-face interaction decreases approximately 70% and is replaced by electronic interactions.

These findings are significant in at least one major aspect: It impacts the outcomes of socialization processes that aims at sharing tacit knowledge through imitation, observation or practice [122]. If the tacit knowledge sharing is jeop-
ardized by physical structures, companies should design socialization strategies with specific purposes to shift the ownership from people to organization [46,126].

4.4.6 Team environment/ settings - C6

This category refers to teams’ environmental set up with respect to people’s knowledge and attitude. It supports continuous assimilation of change by gathering different perspectives for problem solving and performing tasks. Moreover, it facilitates locating knowledge sources. This category supports C5 in the sense that the social collaboration might be enhanced when people with different backgrounds work together and they are aware of each ones’ knowledge.

The knowledge diversity perspectives contribute to the coexistence of different ideas. The goal is to have opposite ideas that can converge to new ones, resulting in new knowledge [122]. On this, P8 comments “I’ve been here quite a while and there are people that are here for two or three years, and I really want a group that is diverse in that sense and have different backgrounds in what they did.”

The convergence to new knowledge within and cross teams is moderated by the combination of interpersonal skills where people not only communicate with each other, but also coexist and interact with different personalities. There are two important aspects in this sense: People’s behavior towards the understanding of the cognitive processes for combining diverse knowledge, and the management insight to apply suitable practices, as remarked by P11: “If you are an extrovert person and you say a lot, it is important that you talk the right things and not just talk. We have more introverts and that is alright. You need to know what their strengths are, so I can use them in the right context.”

The team environments can facilitate or hinder the codification of unrevealed knowledge, which enables the companies to have ownership of the knowledge resource. In domains where skilled people are required to perform an activity with an specific type of knowledge, the codification of the knowledge becomes even more important.

The continuous assimilation of change often relies on the continuous learning awareness, which positively impacts the expansion of people’s knowledge in agile teams. The strategies adopted by the teams vary. They usually depend on the tacitness of the knowledge to learn. The more complex the knowledge is in terms of externalization, the higher the tendency to adopt more socialization between people, such as workshops or building communities for discussions. Prototyping
is also a way of testing the combination of knowledge originated from cognitive processes, as well as formal learning programs.

Finally, knowing what others know is a key element to consider when forming teams and collaborating inside and across teams. When teams are collocated this is facilitated by the constant interaction between people. However, when it comes to larger organizations, for example with distributed development, knowing what are each individual’s competence might be challenging and essential at the same time.

SYNTHESIS: Behavioral software engineering is a recent growing field [104]. The areas explored in this field are closely related to the concepts we have described in this category such as group thinking and team composition [104, 105, 151].

Teams rely on heterogeneity regarding both knowledge diversity and interpersonal skills for effectively solving complex problems, which is part of the software development activities [27]. However, agile teams have a tendency to become homogeneous over time in terms of routines [169].

KM practices that enhance the collective thinking could be implemented to trigger cognitive process for combining knowledge from people. Nonaka et al. [123] suggest that activities that promote creative chaos could stimulate knowledge creation processes, which might reduce the homogeneity within the teams.

Teams could benefit from utilizing these concepts in coordination activities despite the context where development is carried out (locally or distributed). What is important to consider, however, is how these concepts vary in these contexts, for example, in distributed contexts knowing what others know could be managed through ICTs or virtual communities.

In other contexts where knowledge is deeply tacit but essential in terms of sharing from individual to organization level, live interactions are more suitable. Clearly, communication plays an essential role in this regard since the combination of interpersonal skills in a team including, for example, extroverts, and thinking or intuitive people affects the interaction within and between teams [27].

4.4.7 Inefficient utilization of the knowledge resource - C7

Software companies achieve continuous assimilation of change utilizing agile coordination mechanisms, despite the inefficient utilization of the knowledge resource. Even though the existing awareness of the relevance of knowledge
codification, practitioners face difficulties recognizing what knowledge should be codified into an artifact in daily activities and when.

In combination with these difficulties, practitioners codify irrelevant knowledge. This practice occurs mainly through software and databases, which are the most used technologies for codification in the software companies. The result is a meaningless search in database, which is not updated and lacks trust from users. In this regard, P16 comments: “most of the time you get 2000 results and go through, you search. The first 200 items are outdated, and below that is misinterpreted, so not really the right one.”

While focusing on fast delivery through iterations, the inefficiency in terms of KM in coordination activities raises frustration because of recurrent problems that were solved entirely or partially before. Practitioners mentioned that in several cases that they redesign solutions, which was confirmed by P17: “Of course it happens...you can search what others have done in the past, but we have a database that there is thousands of issues, how do you search in the database?”

One important factor that frequently results in knowledge loss is that people’s knowledge gets attention mainly during a staff turnover. In this case, employees with a specific technical knowledge, for example security or streaming, occasionally are requested to codify “what they know” in relation to a topic in a short time. In this process, knowledge is partially lost, which affect other employees’ learning time.

The inefficiency in the management of the knowledge resource also affects the competitive positioning of a software company in a market. In particular cases, where software development companies are market leaders or closely compete with the leaders, the inefficient coordination of requirements engineering activities might motivate the company’s disconnection of external environment. This scenario is characterized by the unawareness of knowledge sources for eliciting new requirements through cognitive processes, commented by P9: “Sometimes it is very hard to see what business value does this bring, short term versus long term. There are one or two layers between the functionality I add to the actual business and selling, so makes quite hard to to know. I go with my gut feeling a lot.”

Finally, through the different coordination mechanisms and activities within ASD, the companies waste substantial time on searching for relevant knowledge that might be already codified or inside people’s minds. In this regard, P3 comments: “The documentation is poor, each team keeps their own repository, and things get worse when you go to other departments.”
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

SYNTHESIS: Rus and Lindvall [141] have mentioned that KM could prevent and mitigate the risks in software development companies related to knowledge loss, lack of knowledge, rework and problem solution redoing, and staff turnover.

The inefficiency concepts related to knowledge resources management have been confirmed by the existing literature [19, 48, 61, 87, 148], but our findings also reveals additional inefficiencies such as redesigning solutions due to unawareness of knowledge sources. In the sections that follows, we expand this discussion with potential implications.

4.5 Discussing implications for KM in ASD

In this section, we list and discuss the most relevant implications raised from the continuous assimilation of change phenomena detailed by CHASM model. We suggest three main complementary strategies to manage the knowledge resource: knowledge codification, knowledge sources, and strategic management.

4.5.1 Knowledge codification

The knowledge codification in ASD has different levels of formality, which is guided by the companies’ documentation strategies and policies. Documentation currently focuses on feature specifications, tests, changes and problem solutions.

Besides the companies’ codification rules, most teams keep what they believe is relevant in repositories. Formal artifacts are commonly software provided by the company. Artifacts adopted informally are selected by the team themselves, e.g., Wikis.

However, the lack of clear knowledge structure and guidance might result in time and economic waste. Practitioners (P1, P3, P9, P14, P16) agree on the importance of keeping knowledge somehow to attend different types of users and applications. Still, the expertise about how to do this does not exist.

The developers within the teams might perceive the codification process as a distraction. Stettina et al. [154] found that the codification activities were given to the least qualified developer, while the rest of the team focused on what they thought was the important, which was writing code. The time pressure molds people’s behavior to focus on very specific features, giving less attention to peripheral but not less important activities [8, 57].
4.5 Discussing implications for KM in ASD

In contrast, another study points out that decision processes related to the cost estimation in future projects might result in miscalculation due to the uncodified knowledge predominantly in the requirements engineering phase [144].

Knowledge codification can be costly when planned and implemented in an appropriate way but it positively affects innovation, economic growth and knowledge creation [10,38].

Cohendet and Steinmueller [38] explain that the use of information and communication technologies (ICT) significantly reduces the costs of knowledge codification activities and also facilitates the dissemination of this knowledge. Additionally, it assists knowledge application into routines.

In agile teams the implementation of ICTs focusing on codifying and also sharing knowledge should pay attention to at least three specific items:

- **Match codified knowledge to people’s knowledge needs.** Teams are working with different items in different development phase, which characterizes a complex environment with several knowledge sources and several potential users. Thus, this aspect includes the range within which the knowledge should be accessible and to whom; design or plan the usage of ICTs as guidance to relevant job-related knowledge [81].

- **Knowledge structure.** It relates to the form of the knowledge when codified into an artifact and how it affects the usability and applicability of the knowledge by the user. According to Hall [76], two main aspects are critical to know how to codify knowledge: understand why people need certain knowledge and how it might be applied.

- **Knowledge update.** To make sure that the knowledge matches the users’ needs it is crucial to update the knowledge already codified [99]. Assimilating changes is a constant in ASD and to manage them different knowledge needs might appear.

It is important to bear in mind that one should not expect that codification is a synonym of ICT. The existence of knowledge codified into artifacts does not guarantee its applicability, but still depends on both social and cognitive aspects that allows people to use the knowledge.

Therefore, it is likely that an appropriate codification process aligned to an effective use of the knowledge [2] can result in reduced time waste and deployment delays by speeding up the knowledge retrieval, however, further studies will need to be undertaken.
4.5.2 Knowledge Sources

Knowledge needs have different levels, sources and goals associated with. They vary from high level goals and strategic directions to specific requirements or source code [141].

From the KM perspective, most ASD companies do not adopt practices aimed at identifying knowledge needs or at supplying them through knowledge sources. As we discussed in the previous section (see to 4.5.1), most knowledge codification activities are guided by company policies for keeping product documentation that are often outdated.

With the exception of the knowledge codified in the artifacts, knowledge needs in ASD contexts are supplied through informal communication between people either inside the teams or cross-teams. However, they might not find sources needed for acquiring the knowledge. They lean towards communicating in a close radius range with people they already know.

From this inconsistency between the knowledge needs and sources, two scenarios emerge: 1) People’s confidence in knowledge stored in artifacts and 2) the unawareness of the knowledge of others.

Practitioners (P14, P17, P18) mentioned they prefer to not consult their database as a knowledge source to search for similar problems/solutions because it is too large and outdated. This phenomenon happens because the recipient’s behavior towards a knowledge source is influenced by how reliable the source is [5, 159].

Reliability also affects cooperation between people as sources of knowledge themselves. The more reliable the source, the more cooperation due to people’s receptiveness [53]. In addition, reduced costs of knowledge exchange is expected [44].

The second implication relates to the unawareness of other people’s knowledge. In small companies where teams are collocated and there is a low number of employees, mostly everyone knows each others competences. It gets easier to find knowledge sources that meet knowledge needs. However, in contexts where the number of teams and employees do not allow this connection, the time for solving problems might increase, and in extreme cases the cost increases in relation to acquisition of external sources of knowledge.

Unanimously in the companies, even though companies have line manager roles or similar, when it comes to day-to-day work, team members are unaware of other people’s knowledge. Another aspect pointed out by P2 during the data collection is that by knowing what the others know, you are likely to plan your
4.5 Discussing implications for KM in ASD

human resources more adequately, since you might know the “load capacity” of each individual.

Corroborating with this analysis, Rus and Lindvall [141] point out five areas of knowledge areas in software engineering that are critical to achieve business goals: knowledge about new technologies, domain knowledge, internal practices and policies, knowing who knows what, and collaboration for sharing knowledge.

There are different possible approaches to manage knowledge sources. However, the main step is the knowledge mapping [150], which includes activities for identifying relevant sources of knowledge to bridge the knowledge gap among people within a company. This mapping can be executed in external knowledge sources such as new technologies and market trends, and internally aiming at bridging the gap between who knows what and where knowledge is missing.

4.5.3 Strategic management

The companies adopting ASD employ and adapt coordination methods focusing on breaking down the work and assigning responsibilities. These practices have little connection to the management of the knowledge resources. Although knowledge has been discussed in several studies in software engineering, it is still an abstract concept. At present, research has not focused on how to apply knowledge as a resource [128].

The main implication from the strategic management point of view is the impossibility of assessing KM effectiveness. To exemplify, product awareness (See to 4.4.3) is an important knowledge resource. Its mismanagement might have implications for writing unnecessary code for example. The practices focused on this aspect in ASD are informal and rely upon individual employees’ perception of the event, while different employees might not have the same insight.

An alternative to verify to what extent knowledge contributes on generating business value by addressing mismanagement issues is to incorporate KM practices into ASD coordination activities and develop strategies to analyze the resources application [126].

Knowledge management practices should be developed with the aiming of achieving a purpose. Also, they need organizational support and stimuli [145]. Leadership plays a fundamental role in stimulating and creating an environment for managing knowledge through people.

Similar to tangible resources, knowledge also needs to be managed to assure that its application is effective. In this respect, leadership can guide companies to actively and dynamically apply knowledge in the daily work by creating the appropriate conditions [124].
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources

Although ASD utilize low hierarchies and shared responsibility in the teams, there are roles in different methods and frameworks that can include leadership responsibilities such as product owners and scrum masters. Besides agile specific ones, common roles found in software companies are also qualified including line managers, project managers and architects.

The articulation of middle managers and alternative leadership roles enable the synchronization of the knowledge goals in the entire company [124]. In each company level, the leadership can mediate processes aiming to enhance knowledge creation, storage, sharing/transfer and application [128].

4.6 Threats to validity

In this section we discuss the validity threats of our study. We follow the guidelines recommended by Wohlin et al. [174], which classify the threats in external, internal, conclusion, and construct.

External validity relates to the possibility of generalization of the results outside the investigation settings. Despite interviewing 18 practitioners from five companies, we are aware that this sample is far from representative to the entire software industry. Still, we aim for analytical generalization [64] by providing as rich contextual information as possible and discussing our findings.

Our strategy to further mitigate threats to external validity was to contact companies that develop software combined with hardware and companies that have pure software as main product. In addition, we also looked for companies in different domains. We believe that this strategy of combining domains, product type and market conditions helped in balancing the sampling to support the generalization of the results.

The internal validity relates to the awareness about other factors that might affect the casual relationship investigated. Our study is exploratory rather than confirmatory which minimizes some internal validity threats.

We recognize that the fact that each execution step of the grounded theory relies on subjectivity of the researcher is a threat. To diminish this, we followed the systematic process described by the grounded theory [41]. The design phase was an interactive process. The authors participated commenting and adjusting the proposal and the data collection instrument. In the data collection phase, the first author conducted the interviews that were further discussed with the second author (see Section 4.3).

The conclusion validity threats are related to concerns that might interfere on drawing accurate conclusions. In our study, we recognize that achieving
consistency in the concepts during the data collection was a threat. To mitigate this threat we decided to collect the data in batches. By adopting this strategy, we executed the coding after each batch and compared to the previous concepts, which allowed the consistency in the concepts generated throughout the analysis (see Section 4.3).

Although grounded theory provides ways for validating the theory generation (see section 4.3), we provided each category with a synthesis discussion, including how the existing literature covers the concepts originated from the analysis. In the synthesis, we discussed similarities and contrasts in both software engineering and other areas literature.

In the construct validity, we discuss possible threats between the research setting and the theoretical construct we explore. One threat we identified was the inadequate preoperational explication of the constructs from different areas such as knowledge management. To reduce this threat, during the interviews we omitted the specific terminologies of KM since it is not common knowledge for the practitioners (see supplemental material A). Later in the the analysis we applied the terminology. For example, when we use knowledge diversity perspectives we are referring to people with different backgrounds inside the team.

To avoid mono-operation [174] bias we interviewed a wide range of people with different roles and working with different agile methods. In relation to the evaluation apprehension, before the interviews started we explained to the practitioners that they would be anonymized and the results would be combined. In addition, we explained that since the study was exploratory, we did not aim to evaluate, but to understand how the phenomenon happened.

By following the systematic process of grounded theory, we identified and gathered 44 KBRs (RQ1) into the Continuous Assimilation of Change Model. The model explains how practitioners utilize the identified KBRs in daily routines. The KBRs support coordination (RQ2) through a business analytic perspective, product systemic reasoning, and ability to codify and transmit knowledge. This have found 44 knowledge-based resources (RQ1), aggregated in six categories. These resources support coordination in ASD with continuous assimilation of change (RQ2).
4.7 Conclusions

Knowledge is recognized as a major resource for software development. However, the lack of understanding about how to manage this abstract concept in real practices might hinder its utilization.

To minimize this abstraction, we examined the knowledge from the resource-based view perspective, which gave us insights about how this resource supports coordination in ASD.

By following the systematic process of grounded theory, we identified and gathered 44 KBRs into the Continuous Assimilation of Change Model. The model explains how practitioners utilize the identified KBRs to support coordination. The explanatory potential of the model was validated through comparison between the categories we established and the existing literature.

The overall results show that the practitioners utilize KBRs in their routines, through social collaboration within teams environment/settings. However, this process is non systematic, which brings inefficiency in terms of knowledge resource utilization resulting in knowledge loss. It can generate negative implications to the course of the software development including meaningless searches in databases, frustration because of recurrent problems, redesign of solutions, and unawareness of knowledge sources.

To utilize the model, it is important that the practitioners notice that similar to any other type of resource, the knowledge-based ones have different levels of importance to each software company. As a starting point, we suggest that practitioners prioritize critical KBRs and develop strategies to manage them. The strategies could be thought of in terms of the three main implications that we described in Section 4.5: knowledge codification, knowledge sources, and strategic management.

Regarding future research, we foresee as relevant aspects to explore two main issues. The first is related to the lack of effectiveness measures to KM; how to define metrics to evaluate the outcomes of KM strategies. The second is related to the balance between a completely informal and inflexible environment in terms of communication and codification; how to define this balance and how does this affect the agile flow?

These findings, while preliminary, suggests that the understanding of the role of knowledge as a resource can promote changes to the way that agile methods have been applied. This might happen especially due to a rethinking about what documentation is and its relevance as a resource that, if well managed, can avoid the time and knowledge waste.
4.8 Appendix
Continuous Assimilation of Change in Agile Software Development: An empirical study on the role of the knowledge-based resources
Chapter 5

A Method to Evaluate Knowledge Resources in Agile Software Development

This chapter is based on the following paper:


5.1 Introduction

Software development is knowledge-intensive, and knowledge is one of the key resources of many development organizations. At the same time, the software development landscape frequently changes, demanding from development organizations to learn new skills, apply new business models, and development paradigms [172]. This pressure to obtain new knowledge is then combined with the continuous degradation of the existing knowledge into obsolescence [35].
Moreover, many development organizations utilize tacit knowledge rather than explicit knowledge. Tacit knowledge is recognized as the most important resource in an knowledge-intensive activity such as the design of a new product or the code itself [12, 70, 73, 153]. The domain knowledge and rationale behind software design decisions often remain tacit and undocumented.

Organizations manage their knowledge resources through codification and personalization strategies [78]. Codification relates to the activity of storing knowledge into artifacts (e.g., Wiki-based tools) when the knowledge nature tends to be explicit; personalization relates to socialization activities to communicate knowledge (e.g., team meetings and discussion groups) when the knowledge nature tends to be tacit.

Most of the Knowledge Management (KM) practices used in the Agile Software Development (ASD) context follow the personalization strategy, which rely on people socializing for sharing knowledge [128]. This strategy, while effective regarding flexibility and agility, might not scale for several teams [52]. In this scenario, knowledge can be lost or not reused by other teams. On the other hand, databases used as a codification strategy can also be filled with outdated and irrelevant knowledge [99].

Although software development organizations utilize several KM practices to retain and share knowledge, what is still missing is a way to evaluate the criticality\(^1\) of a knowledge item [34, 54, 90, 98, 128].

To address this gap, we propose KCEM. Its objective is to support companies to systematically retain knowledge in ASD contexts. It also contributes to avoid the accumulation of unnecessary artifacts. We designed KCEM to be lightweight and easily applicable by practitioners.

In this study, we report the initial results of a larger and longer-term research, which is to develop a scalable KM solution for companies adopting ASD. Our initial results refer to the developed method to evaluate the criticality of knowledge items in ASD contexts. Thus, we answer the following research questions in this paper:

- **RQ1**: How to evaluate knowledge criticality in agile software development contexts?
  - **RQ1.1**: What aspects of criticality should be considered?
  - **RQ1.2**: Which rubrics should be applied to the aspects?

The main contributions of this paper are:

\(^1\)The degree to which a knowledge is essential to what it has being applied.
5.2 Background and Related Work

- A method to support the evaluation of a knowledge item’s criticality.
- Preliminary results of lab and static validations.
- The discussions presented as lessons learned.

This paper is organized as follows: In Section 5.2, we present a brief background along with the respective related work. In Sections 5.3, 5.4, 5.5 we describe respectively our research design, the proposed method, and the preliminary results of the method validation. In Section 5.6 we discuss lessons learned on our validation. Finally, in Section 5.8, we present our concluding remarks and view on future work.

5.2 Background and Related Work

Knowledge is a concept extensively discussed in the literature in terms of nature and understanding [73]. Since this discussion is not our main purpose, we adopt the a simple knowledge definition, based on Hislop [84], as the collection of information that provides guidance.

The two types of knowledge are: tacit, which is individual deriving from experiences, values and routines; and explicit, which is already systematized in formal language including guidelines, instructions or books [124].

Companies manage their knowledge resources through two main strategies: codification and personalization [78]. Codification relates to the activity of storing knowledge into artifacts (e.g., Wiki-based tools) when the knowledge nature tends to be explicit. It has a strong focus on the use of information technology mainly because they aim at reusability and broad access.

Personalization strategies relates to socialization activities that focus on person-to-person to communicate knowledge (e.g., team meetings and discussion groups) when the knowledge nature tends to be tacit. The strategies do not prioritize information technology since human interaction is the main objective [78].

Knowledge-based resources may have different levels of importance for different companies [73]. The decision on to retain the knowledge and how, depends on the criticality of the knowledge. There are few studies available in the literature which explore knowledge evaluation methods [60, 162, 163]. They focus mostly on generic criteria, which requires several adaptations to comply with the particularities of the ASD context.
Tseng and Huang [163] used modeling knowledge requirements to create KM systems. The approach is limited by the context where it is used and also restricts the evaluation to explicit knowledge. However, it provided us with insights regarding the utilization of quadrants to classify the knowledge in terms of criticality.

Ermine et al. [60] utilized a generic grid to evaluate knowledge mostly applied in French companies. It is a private method, thus, not available in the literature. Even so, we still got inspired by its thematic axes to create our categories and rubrics for our method, considering the team feature, which is a particularity of most ASD methods.

Based on these studies, as well as on the output from a preliminary seminar with the company, we created the categories and rubrics for analyzing knowledge criticality and the strategy for the classification (see Section 5.4).

5.3 Research Design

We conducted an improvement case study [140] to develop and evaluate KCEM, and improve the decision regarding knowledge retention. Furthermore, this research follows the guidelines for technology transfer between industry and academia proposed by Gorschek et al. [71]. The case and unit of analysis is Ericsson2, a Swedish company that develops telecommunication solutions.

The aforementioned model suggests a set of seven flexible steps for researching in collaboration with industry. We focus on the five first steps of the model to introduce the initial results of our research. In the first two steps, we conducted a workshop at Ericsson, which had two main purposes: i) present the results of a previous study regarding knowledge-based resources in ASD; and ii) elicit the potential issues that future research could address. As results of this workshop, we identified potential improvement areas, agreed on a research agenda, and prioritized the areas.

Data collection. To evaluate KCEM, we collect data in two phases of the technology transfer model: lab validation (step 4) and static validation (step 5). In both cases, we gather feedback from the participants regarding the list of questions for the defined rubrics, as well as the degree of agreement between the rubrics and the real world.

Data analysis. Since we collect feedback in both validation phases (detailed in 5.5.1 and in 5.5.2), we refine KCEM accordingly for the next phase. The

2www.ericsson.com
issues raised during the validation are discussed in the form of lessons learned in section 5.6.

5.4 The knowledge criticality evaluation method - KCEM

We define knowledge criticality as the degree to which a knowledge item is essential to what it has being applied. The method evaluates Knowledge Items (KIT), which we define as the particular unit of knowledge. A KIT can be different things such as a process and its associated knowledge. A KIT can be either tacit or explicit, e.g., coordination and interpersonal skills or knowledge stored by developers in wikis.

To illustrate an example of KIT, let’s consider the Build Master. This role may have many different KITs, such as Expertise in git\(^3\), proficiency on the company release process, and proficiency on the process of fixing software faults.

Criticality is divided into two categories, relevance and scarcity, which we define as follows:

- **Relevance** - potential for satisfying a particular need, e.g., within a team or larger agile setups.

- **Scarcity** - relates to the extent that a KIT is readily obtainable, e.g., the number of experts that has a specific skill.

Each category has a set of rubrics (see Table 5.1), which can be used by an individual or by a group (in a planning poker style). Each rubric is a statement/question whose answers represent a rating (e.g. agreement, likelihood, and importance) relationship, thereby we could treat them as Likert items, with values ranging from 1 to \(\max\), with \(\max \in \mathbb{N}\), e.g., a scale with five levels, one could answer 1, 2, 3, 4, or 5.

Since we have Likert items with different levels, the answers of each rubric \(r\) are normalized to the range \([0, 1]\) by applying a general formula \(\text{score}(r) = \frac{n-1}{\max(r) - 1}\), where \(\max(r)\) is the maximum value for the scale related to the rubric \(r\), and \(n\) is the actual answer. This normalization allows the calculation (sum of the values of all rubrics of each category) of scores for both Relevance and Scarcity, which facilitates interpretation of the measurements. Since both categories comprise five rubrics, both scores range from 0 to 5 and, in order to

\(^3\)A broadly used software for versioning control
Table 5.1: Rubrics for evaluating knowledge criticality

<table>
<thead>
<tr>
<th>Category</th>
<th>Rubric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevance</td>
<td>Potential to cause delay</td>
<td>Relate to dependency of this specific knowledge to support different tasks/activities that might get delayed if the knowledge is not available</td>
</tr>
<tr>
<td></td>
<td>Obsolescence</td>
<td>Potential for the knowledge item become obsolete</td>
</tr>
<tr>
<td></td>
<td>Adaptability</td>
<td>Relates to the extent that the knowledge item has potential to be adjusted to different conditions</td>
</tr>
<tr>
<td></td>
<td>Dependency</td>
<td>Dependency on the knowledge to perform the tasks of a specific or more roles</td>
</tr>
<tr>
<td></td>
<td>Potential cross-benefit</td>
<td>Several teams would benefit from having this knowledge available</td>
</tr>
<tr>
<td>Scarcity</td>
<td>Available experts</td>
<td>Relates to the number of experts, which owns a specific knowledge, and could supply the need for it in case of, e.g., a turnover</td>
</tr>
<tr>
<td></td>
<td>Originality</td>
<td>The knowledge is somehow original. It was not owned by the company before</td>
</tr>
<tr>
<td></td>
<td>Difficulty of externalization</td>
<td>Relates to the easiness of representing this knowledge item in an artifact (e.g., guidelines, instructions)</td>
</tr>
<tr>
<td></td>
<td>Seasonal knowledge</td>
<td>Relates to knowledge that is used to specific occasions, not frequently</td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td>Relates to how complex this knowledge is when comparing to the ones used in daily activities</td>
</tr>
<tr>
<td></td>
<td>Staff turnover</td>
<td>Relates to the rate of staff turnover</td>
</tr>
</tbody>
</table>

define the four quadrants, we use 2.5 as middle point. Note that we use radar charts to facilitate the visualization of the results of the rubrics of each category.

The result of the calculated scores is a pair \( \text{crit} = (\text{rel}, \text{sca}) \) visualized in a 2-dimensional chart (see Fig. 5.1), which has four quadrants for recommendation categories for managing the evaluated knowledge. The quadrants are described as follows:

- **Q1 - Slight criticality**: The KIT has low scores for both scarcity and utility, i.e., it has low complexity and probably will be readily available when needed.

- **Q2 - Low criticality - Vigilant**: The KIT’s utility is low, but high scored in scarcity, which might be challenging to acquire it quickly. It might not have, e.g., homogeneous importance across teams or to a development process. Thus, the recommendation for this knowledge item is to make efforts to, e.g., duplicate or make it redundant on occasions where and when it is needed.
• **Q3 - Medium criticality - Handy**: The knowledge item is mostly available and important. The recommendation is to facilitate accessibility through efficient search processes, either through socialization or codification processes. For example, codify knowledge in artifacts (instructions, guidelines) or, in case of tacit knowledge, make the source visible.

• **Q4 - High criticality - Strategic**: The knowledge item is crucial and potentially contributes to continuous readiness to absorb unforeseen changes in agile contexts. Thus, it is strongly recommended to establish knowledge retention practices for this knowledge item.

![Figure 5.1: Knowledge criticality](image-url)
5.5 Validation

We performed a two-step validation. We briefly explain the lab validation, and discuss the static validation in the following sections.

5.5.1 Lab Validation

Five software engineering Ph.D candidates participated in the lab validation. They utilized the method to evaluate KITs related to their context, including knowledge about how to narrow down research questions and report travel expenses.

We collected early feedback from the students regarding usability of the method and about the clarity of the questions.

5.5.2 Static Validation

In the static validation, we utilized the company’s documentation that described the development processes to collect the critical knowledge associated with each process. This step was performed together with a practitioner from the company.

Afterward, a representative of each process utilized the method and gave us feedback. In total, four line managers that are responsible for each process utilized the method. Each of them evaluated one KIT, except one that felt comfortable also evaluating the fifth one.

We selected five critical knowledge items to be evaluated by the practitioners participating in the workshop. Each KIT belongs to a different development process from one of the Ericsson’s products. They are:

- KIT1: How to design test cases
- KIT2: Do trouble report handling
- KIT3: Do emergency pack
- KIT4: Start new technical product group
- KIT5: Pre-product customization development

The practitioners evaluated the KITs in approximately 10 minutes and for each of the items, we have generated radar charts (see Fig. 5.2) to supplement the knowledge criticality chart, see Fig. 5.3.
5.5 Validation

Besides the evaluation of the questionnaire, we collected feedback regarding the degree of agreement of the method to the real world. The group discussion aimed at validating the following items:

**To what extent the method helps on recommending how to manage the knowledge item.** The practitioners agreed to a large extent that the method is good to evaluate KITs under the KM perspective. However, as a supplement to the method they recommended that at the same time that they evaluate the criticality, they could also answer about the current status of that KIT in terms of management.

**Does the method provide a somewhat different perspective about the knowledge item?** The practitioners reported a good experience using the method in two main aspects: 1 - The visualization of the method gives a different perspective about the KIT and facilitates comparing the criticality of different KIT within the same or different processes; 2 - The easiness of use was also mentioned by the practitioners, which corroborate with one of our objective of developing a lightweight method to easy application.

**To what extent the results seems accurate.** Even though we were not able to breakdown the processes in detail, the practitioners confirmed the accuracy of the results during the discussion. For example, while presenting the results, the

Figure 5.2: Rubrics results for the KITs
two practitioners that evaluated KIT1 and KIT2 confirmed that the scarcity for those KITs are explained by the fact that the knowledge to perform those tasks is quite available for when it is needed, so its scarcity is low, and the relevance is quite high for the development processes.

5.6 Lessons Learned

Overall, the method was received as quite simple to apply, with some difficulties to gather the KITs. The most important lessons learned are the following:

Lesson #1 Gathering the KITs may be time consuming depending on how large is the analyzed process.

We utilized the Ericsson’s high level document that depicts the main development processes and a few sub-processes to validate the method. However,
we plan to utilize cognitive task analysis (CTA) for dynamic validation as it offers a more complete picture of the knowledge required to accomplish each task in the sub-processes. CTA is a set of methods that aim to capture and understand how cognition support people in accomplishing their tasks [43]. By applying this technique, it is possible to map tasks in companies that do not have a structured and detailed process. The second lesson learned:

<table>
<thead>
<tr>
<th>Lesson #2</th>
<th>The level in which the KIT is evaluated has to be defined upfront to avoid confusion regarding the context of the item.</th>
</tr>
</thead>
</table>

We noticed that the development processes evaluated is quite extensive due to the size of the product, and some practitioners were confused to consider which context they should evaluate. For example, when considering the rubric “dependency”, they should evaluate considering the task in relation to the deliverable, process or product in general? We were able to discuss and define the context this during the validation, however it is an important aspect to consider for researchers and other practitioners who want to apply the method.

<table>
<thead>
<tr>
<th>Lesson #3</th>
<th>The relevance scores tend to be biased because practitioners believe that all knowledge is highly relevant.</th>
</tr>
</thead>
</table>

The practitioners confirmed that they tend to overrate their tasks and its knowledge associated. The way we phrase the questions, specially regarding relevance, can provide unreal results.

<table>
<thead>
<tr>
<th>Lesson #4</th>
<th>Allowing the method to have multiple entries, practitioners can evaluate the same item simultaneously.</th>
</tr>
</thead>
</table>

There might be disagreements among practitioners when evaluating the KITs. However, we do not consider this a problem to our method. Since we adopt the concept that knowledge is socially constructed [123], allowing multiple entries to the same KIT will promote relevant discussion towards a better strategy to manage the KIT.
Lesson #5 The evaluation should be followed by questions if this KIT has already strategic plan to be managed.

One of the improvements that can increase the utility of the method during the discussion of the results is to add extra questions aiming at evaluate the current practice to manage the KIT. Thus, when visualizing the results, the practitioners can also see the current status of the KM practice related to the KIT evaluated.

Lesson #6 Having arbitrary intervals raise the need for analyzing the rubrics individually.

We chose to use arbitrary intervals to classify the KITs in method. However, we recommend that when the result falls in the border to other quadrant, practitioners should examine carefully rubric by rubric in the radar charts. For example, the KIT4 (see Fig. 5.3) is close to Q2. In this case, the KM practice adopted to manage the KIT can vary dramatically. The company might spend many resources to retain knowledge that it is not widely critical or the contrary, retain the knowledge locally, e.g., in one team, when it is critical to several parts of the company.

5.7 Validity Threats

We follow the guidelines recommended by Runeson and Höst [140], which classify the threats in external, internal, reliability, and construct.

External validity relates to the extent that the results can be generalized. Although the results and lessons learned are limited to Ericsson’s case, we believe that the rubrics are generic enough to be utilized in other software development companies adopting ASD.

Internal validity relates to the awareness about other factors that could affect the phenomenon investigated. The experiences reported could be different if more people from the same process have used the method. However, this threat does not represent a high risk because the method does not seek agreement, but allows discussion with different points of view instead, so a better KM practice is adopted in the end. Besides, we have discussed the findings with representatives of each analyzed process.
Reliability relates to how repeatable is the study. The results of the application of the method will vary from company to company, especially because they will probably identify different knowledge items as input. However, to make the research reproducible, we detailed all the steps carried out in our investigation.

Construct validity relates to the extent that the measures utilized reflect what both researchers and the population investigated have in mind. To mitigate this threat, we explained the meaning of each rubric and category before applying the method, and also discussed the obtained results.

5.8 Conclusion

In this paper, we proposed a method for evaluating the knowledge criticality of knowledge items in ASD contexts. We have conducted a preliminary validation of the method, which was divided in two parts: lab validation and static validation. Both validations show promising results.

Considering that the main goal of our research is to support companies, the results of the static validation are of special interest. The practitioners who used our method highlighted three main advantages: easy to understand and use, provide a different perspective on a KIT by visualizing the criticality chart, and reduce the level of abstraction associated to a knowledge subject area.

The findings of this study have important implications for future KM practices including changing the way we perceived knowledge towards a resource and reducing the level of abstraction of the topic.

This research is part of an ongoing research at Ericsson. Our long term objective is to pack a scalable KM solution for companies adopting ASD. As the next steps in this research, we aim the following:

- Perform the two remaining steps of the technology transfer model. Regarding the dynamic validation step, we plan to conduct through larger pilots involving more teams. In relation to the final step, Release the solution, we plan to roll out the method at Ericsson.

- Develop an approach to help managers relate the results of using our method and existing competence in the company.

- Automate partially or entirely the criticality analysis of KITs. This might help to identify changes regarding relevance and scarcity in a more timely way.
A Method to Evaluate Knowledge Resources in Agile Software Development
References


