KNOWLEDGE MANAGEMENT PRACTICES IN DEVOPS

Soha Solouki

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Master of Science in E-Business Technologies

Supervisors: Dr. Umar Ruhi & Dr. Lysanne Lessard

Telfer School of Management
University of Ottawa

Abstract

DevOps, a portmanteau of Development and Operations, is the collection of principles and practices that try to improve cooperation between IT Development and IT Operations teams in the software development domain. The DevOps paradigm, thus, promises to overcome the traditional boundaries between development and operations teams and to improve collaboration across teams through a culture that is conducive to shared goals and accountability.

Responding to the recent call for a better understanding of DevOps Knowledge Management (KM), this study aims to explore the role of knowledge management in advancing DevOps performance outcomes. Toward this, the study adopts a practice perspective of KM, and aims to answer the following research questions: 1) What are the enablers of KM practices in DevOps teams? 2) What are the distinctive characteristics of KM practices that underpin positive DevOps performance outcomes?

Using an inductive research design and qualitative data collection and analysis procedures, this study followed a multiple case study approach, and collected and analyzed data from nine in-depth interviews with DevOps professionals across three organizations. Using grounded theory coding procedures, an emergent theoretical model of DevOps KM is presented and discussed, along with various propositions that outline how DevOps teams acquire, capture, share and apply knowledge, and how their KM practices can drive positive DevOps performance.

Key insights from this study indicate that technology leaders need to foster greater awareness about the significance of KM in DevOps teams. This can be done by highlighting challenges associated with a lack of effective KM practices, and best practices followed by other companies. Furthermore, DevOps teams should adopt a mix of people-centered and technology-centered KM practices that enable effective personalization and codification of knowledge. Lastly, DevOps managers need to encourage alternative-bridging KM practices through regular use of KM tools and features within DevOps technologies while investing in dedicated knowledge sharing platforms.

Through a discussion of the enablers of KM practices in DevOps; typical configuration of people-centered, technology-centered, and alternative-bridging KM practices in DevOps; and the linkages between KM practices and DevOps performance outcomes, this study aims to contribute to the extant research literature on DevOps KM, and provide practical guidelines for institutionalizing KM practices that can support the fast-paced nature of DevOps teams.

Acknowledgement

I wish to thank the following people whose help was a milestone in the completion of this project.

I would like to pay my special regards to my supervisor, Professor Umar Ruhi, who defined the path of my research generously and provided access and introductions to useful resources for my research. I wish to express my sincere appreciation to my supervisor Professor Lysanne Lessard, whose insight and knowledge has been invaluable throughout this research project.

I have been blessed with a very supportive family. My parents whose constant advice and great love kept me going on in my life. As for my husband Amirabbas, I would like to thank you for your unfailing love, encouragement, and inputs in every moment of my study journey.

I am also grateful for receiving financial support from various sources to complete this thesis, including research assistantships from the supervisors, University of Ottawa's Electronic Business Technologies matching funds, and Telfer School of Management's Student Thesis Research Grant.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Research Motivation

Given that effective implementation of Knowledge Management (KM) leads organizations to financial and non-financial competitive advantages, research on this topic has grown at a rapid rate (Cerchione, Esposito, & Spadaro, 2016; Delen, Zaim, Kuzey, & Zaim, 2013). In their seminal research on KM, Alavi and Leidner (2001) highlighted four core processes in KM as creation, conversion, storage, and application of knowledge, and posited that firms can enable each of these processes through the appropriate management practices and technology systems practices. Scholars have also underscored the role of these core KM processes in improving a firm's economic, market, technical, human, and organizational performance (López-Nicolás & Meroño-Cerdán, 2011; Obeidat, Al-Suradi, Masa'deh, & Tarhini, 2016; Zheng, Yang, & McLean, 2010). Moreover, KM processes have also been shown to positively impact customer satisfaction as well as internal human- performance (Egbu, Hari, & Renukappa, 2005; Gholami, Asli, Nazari-Shirkouhi, & Noruzy, 2013).

Despite these important insights, we know little about KM processes and practices in agile work contexts. In the context of technology firms, agile practices refer to a socio-technical approach that focuses on cross-functional collaborative team-oriented development rather than rigid software development being undertaken by the technologists (Jabbari, Ali, Petersen, & Tanveer, 2016). In the same vein, Davenport (2010) argues that in contrast to the traditional detailed engineering methods, agile methods are people-oriented, welcome change, and tend to complete discrete project stages in short spans of time. Previous research directs scholarly attention to further investigate agile work methods and their implications for KM (Balijepally, Mahapatra, & Nerur, 2006; Cao, Mohan, Xu, & Ramesh, 2009; Fruhling & De Vreede, 2006). Agile methods have become popular in recent years. For instance, there is evidence of a shift towards a combination of agile and traditional knowledge sharing methods by using extensive documentation and pair programming[1] concurrently (Cram & Marabelli, 2018). Over 70% of companies apply social networks to address team-oriented development and to manage knowledge (Kubátová,

[1] Pair programming is an agile software development technique in which two programmers work together at one workstation. The two programmers switch roles frequently to write code and reviews each line of code (Kim et al., 2016).

2013b; Oseledchik, Ivleva, & Ivlev, 2018). Following recent studies, a deeper understanding of agile KM practices embedded in the agile workplaces such as Development and Operations (DevOps) teams is needed.

DevOps is the collection of principles and practices that try to actively improve cooperation between IT Development and IT Operations teams in the software development domain (Jabbari et al., 2016). The Development team's objective is to develop and launch new features requested by users. The Ops team is tasked with ensuring that users have a reliable product. Erich, Amrit & Daneva (2014a, p. 8) define DevOps as "*a development methodology aimed at bridging the gap between Development and Operations, emphasizing communication and collaboration, continuous integration, quality assurance and delivery with automated deployment utilizing a set of development practices.*" The DevOps paradigm promises to overcome the traditional boundaries between development and operation teams and to improve the common culture and shared goals (Davis & Daniels, 2016; Erich, Amrit, & Daneva, 2014b; Nielsen, Winkler, & Nørbjerg, 2017).

While research on KM practices in the context of DevOps is still in its infancy, some researchers have contended that traditional KM processes for knowledge creation, storage, retrieval, and application are not adequate for such agile work environments (Cram & Marabelli, 2018). For example, research shows that 90% of firms with agile software development (ASD) practices prefer to forego heavy document-driven approaches to knowledge sharing, and instead create and share knowledge through a combination of technology and social means, including discussion forums and stand-up[2] meetings (Singh, Singh, & Sharma, 2014). Recent research has attempted to situate these KM practices in DevOps by focusing on interactions (Erich et al., 2014a) and knowledge conversion (Nielsen et al., 2017) between Dev and Ops teams. Therefore, the significance of KM is well-known, though empirical research on applications and best practices remain scarce in the context of an agile work environment such as DevOps.

This gap in the literature is even more pressing considering that DevOps has idiosyncratic culture, practices, and characteristics that might require developers and operations to use unique KM approaches while working in a collaborative environment. For instance, both DevOps and agile practices require continuous integration and testing. However, DevOps further advance this

[2] Stand-up meetings in which the team comes together for short daily meetings, they discuss what they have completed, what they are working on and any issues that are blocking the work.

practice in organizations by continuously monitoring the performance of the system and by presenting results in automated dashboards (Davis & Daniels, 2016; Jabbari et al., 2016; Sharma, 2017). The virtual wiki-based tools, communication networks, and real-time dashboards are examples of the application of social technologies for continuous monitoring, collaboration, and communication in DevOps teams (Bang, Chung, Choh, & Dupuis, 2013; Kim, Humble, Debois, & John, 2016). Such unique DevOps practices merit more attention in the context of KM in that they help practitioners and researchers to explain how certain KM practices can support continuous development and integration in DevOps.

Responding to the recent call for a better understanding of DevOps knowledge management practices and processes (Nielsen et al., 2017), this study aims to explore how knowledge management studies might usefully be leveraged to advance DevOps. Considering that "Knowledge management as a concept is not directly tied to technology; rather emerging technologies provide a means of enabling more effective knowledge management" (Alavi & Leidner, 2001, p. 17), we hope that the findings of this study will help describe the importance of KM in DevOps, and also provide suggestions on how to best integrate KM practices with DevOps procedures and tool-chains.

1.2 Research Purpose and Question

The purpose of the study is to gain a better understanding of Knowledge Management (KM) practices in Development and Operations teams (DevOps). Specifically, the research aims to investigate how DevOps teams apply KM practices for improved business performance. The leading research questions of this study are accordingly:

1) What are the enablers of KM practices in DevOps teams?

2) What are the distinctive characteristics of KM practices that underpin positive DevOps performance outcomes?

Research findings from this study can potentially help explain the role of KM practices in enhancing agility and innovation related business performance outcomes.

1.3 Overview of Methodology

This study employs a multiple case study approach (Eisenhardt, 1989; Yin, 2017) to compare and contrast different KM practices and technologies in various DevOps teams. Given that research on KM practices in the context of DevOps is still in its infancy (Nielsen et al., 2017), a case study is an appropriate approach to extend theory for this underdeveloped research context (Eisenhardt, 1989). As such, our multiple case study approach is important in aiding our understanding of how DevOps acquire, capture, share and apply knowledge effectively.

To address the research questions, first, a literature review was conducted to identify the role of knowledge management practices as drivers of DevOps principles and performance. Based on this literature review, an initial conceptual framework was developed to guide data collection and data analysis efforts.

To develop an understanding of KM practices that support agile business strategy and operations, this study recruited participants from three organizations that use DevOps in their IT function. We interviewed personnel in the following roles: DevOps Development Personnel, DevOps Operations Personnel, and other pertinent Non-DevOps Middle-Management Personnel. Before starting the main interviews, two pilot interviews were also conducted with DevOps professionals from a different organization. These two in-person interviews were important to improve understanding of the current state of DevOps in the industry, to validate our research questions, and develop our final interview guide and questionnaire.

Drawing upon grounded theory (GT) coding procedures to analyze nine in-depth interviews of DevOps professionals, this study presents an emergent model that highlights the characteristics and effects of KM practices in a DevOps context. It also formulates propositions that summarize key findings, provide answers to our research questions, and warrant further empirical research in future research.

1.4 Findings

Analysis of data obtained from the interviews suggests that many DevOps teams are aware of the role of KM in improving DevOps engagement and in improving performance outcomes. Our results highlight a variety of people-centered and technology-centered KM practices that are

pursued by DevOps teams, and this study was able to link different configurations of DevOps KM practices to specific DevOps performance outcomes.

1.5 Contributions

Research findings from this study can potentially help researchers and practitioners explain the role of KM practices in DevOps teams. The insights from this research could be used to explain how KM practices related to acquisition, capture, sharing and application of knowledge manifest and evolve over time in DevOps work contexts. Through a survey of three case study organizations at different stages of DevOps size, we aim to identify key success factors for KM that lead to positive DevOps outcomes and business performance. The findings of this study could refine our understanding of KM practices and processes in agile work contexts, especially in DevOps teams. We hope that the findings of this study can help practitioners and researchers to focus more on the role of KM practices as enablers of DevOps principles and performance.

1.6 Summary of Chapters

The rest of the thesis is structured as follows. In the next section, we will review the extant research on key themes of this study: DevOps and KM. More specifically, this section focuses on definitions and components of KM and DevOps, and thus allows readers to identify the importance of DevOps and KM in general, as well as in the software development context. Drawing on practice theory, this section defines the conceptual foundation for this study

After developing an understanding of these topics, section 3 introduces the methodology of this study in more detail to show why we deliberately chose the inductive approach, and how propositions were formulated about the role and characteristics of KM in DevOps. In doing so, this section presents the research setting and design, data collection, data analysis, and coding methods. Moreover, the research validity criteria and procedures are discussed in this section.

Section 4 presents the theoretical propositions by demonstrating the results of this study related to each case. Relying on the activities of DevOps teams in 3 organizations, propositions are developed which are consistent with the data obtained from our interviewees. We will explore the KM practices in DevOps teams by conceptualizing a model that tries to make a relation between the components of KM practices, DevOps principles and performances.

Following the above sections, section 5 attempts to compare the result of this study with the current body of knowledge and situates the results in previously discussed KM practices in DevOps. This section will conclude with limitations, contributions, and implications for future research path with respect to the research question of this study.

2 Literature Review

2.1 Knowledge Management

The review of KM literature was undertaken to identify the importance of KM, its definition, characteristics, components, and practices.

2.1.1 KM Definition and Components

It would be worthwhile to look at prior definitions of KM in this section, as KM is one of the main components of this study.

The knowledge management literature has documented that the purpose of KM is to manage both tacit (intangible) and explicit corporate knowledge (e.g., the technical data, documents, software, and policies), and to determine who knows what in order for knowledge to be used anywhere at any time (Alavi & Leidner, 2001; Gupta et al., 2000; Nonaka, 1994; Nonaka, Toyama, & Konno, 2000; Nonaka & von Krogh, 2009). Researchers and practitioners consider knowledge as the most important resource of the firm that can lead to competitive advantage and effective performance of the organization (López-Nicolás & Meroño-Cerdán, 2011; Nonaka, 1994; Obeidat et al., 2016; Zheng et al., 2010).

Scholars broadly categorized KM components into three classes: people, processes, and technology (Bhatt, 2001; Bhojaraju, 2005; Marling, 2004). People or employees (agents) who possess the knowledge play a key role in sharing knowledge and collaborative activities (Tavakoli & Schlagwein, 2016). The process is about standard processes for content management, retrieval, document best-practices, etc. (Alavi & Leidner, 2001; Nonaka, 1994). Tools and technology facilitate the above knowledge sharing processes by supporting the workflow, document management, fast knowledge-sharing (Alavi & Leidner, 2001, 1999; Davenport, 2010).

Importantly, all components listed above are required for effective KM, and by focusing exclusively on one of these KM components, organizations can fail to sustain competitive advantage. As stated by Bhatt (2001, p. 75), “putting too much emphasis on people or technologies is not sufficient; rather, management must revisit the interaction pattern between technologies, people, and the techniques people employ in using these technologies.” For instance, although more than 70% of companies apply social and information technology tools (e.g., intranets, groupware, and databases) to support knowledge-sharing (Kubátová, 2013b), without a friendly

social environment (non-IT issues), collective knowledge may not develop in the organization (Cabrera & Cabrera, 2002; Egbu et al., 2005). Bhojaraju (2005) presents six aspects of knowledge assets that need to be considered while implementing KM in an organization: stakeholder relationships (e.g., contracts and partnering agreement), Human resources (e.g., skills and commitment), physical infrastructure (e.g., communication technology), culture (e.g., organizational values), practices and routines (e.g., formal or informal process manuals with rules), intellectual property (e.g., patents, brands).

There is a wide range of factors that could affect KM. For instance, each organization has its own culture which directly impacts how the company manages its knowledge (Alavi, Kayworth, & Leidner, 2005). Culture represents the end goals, important understandings, norms and practices, commonly held beliefs, and customary practices of the firm (Bell DeTienne, Dyer, Hoopes, & Harris, 2004; Lin, 2014). Culture has been known as one of the strongest predictors for effective knowledge management practices (Bell DeTienne et al., 2004; Detlor et al., 2006; Joshi, Parmer, & Chandrawat, 2012). For example, Milosz and Milosz (2010) highlight the cultural issue and claim that most of the SMEs in Poland suffer from lack of successful implementation of knowledge management systems. However, organizational culture that is characterized by trust and cooperative involvement of employees can lead organizations to effective KM (Bell DeTienne et al., 2004). Routine organizational practices (e.g., the way people answer the phone and report) should align with KM objectives because "if company practices do not promote the free transfer and sharing of accurate and valuable knowledge, then the culture is a major impediment to KM initiatives" (Bell DeTienne et al., 2004, p. 29).

Lin (2014) develops a KM model which defines two specific KM stages affected by various factors: adoption and implementation. The study highlights that factors are affecting KM adoption such as IT effectiveness, top management support, and reward system. On the other hand, factors affecting KM implementation are IT effectiveness, top management support, sharing culture, and competitive pressure. Interestingly, IT support is the most significant antecedent during KM adoption (Evangelista, Esposito, Lauro, & Raffa, 2010; Lin, 2014), while KM implementation stage is mainly affected by sharing culture (Lin, 2014). Other than the organizational culture, Ruhi and Al-Mohsen (2015) focus on national culture and claim that practitioners are obliged to consider this factor while implementing knowledge sharing activities.

Regarding human factors, the knowledge creation process is affected by leaders' innovativeness and creativity (Yeow, Chua, Wee, & Chua, 2013). For instance, rewards and selective incentives can increase the benefit of contribution and encourage employees to exchange their ideas and enforce collective knowledge sharing through open debate (Cabrera & Cabrera, 2002; Yeow et al., 2013). Knowledge sharing processes are enabled by awareness of roles, mutual respect and the level of trust among employees; the proximity of employees and the openness of the owner to share knowledge enforce the knowledge re-use (Yeow et al., 2013). Therefore, without the effective engagement of human resources, KM processes may not develop within an organization.

Additionally, the technological factor has a crucial impact on the implementation of KM (Alavi & Leidner, 2001, 1999; Cerchione et al., 2016). ICT is the backbone of knowledge creation, knowledge dissemination, and knowledge utilization, and thus IT expertise is one of the main drivers of web knowledge sharing (WKS) (Soto-Acosta, Colomo-Palacios, & Popa, 2014). Virtual and physical networks together with time availability enable organizational members to exchange their knowledge within an organization (Chen, Chang, Tseng, Chen, & Chang, 2013). In addition to the cultural, social, and technological issues, the financial issues need to be tackled before the implementation of KM, specifically in SMEs (Nunes, Annansingh, Eaglestone, & Wakefield, 2006).

To investigate the entire KM components and related factors, scholars also attempt to highlight the common KM barriers (Anand, Kant, & Singh, 2013; Joshi et al., 2012; Miłosz, 2010). Knowledge sharing barriers (KSBs) refer to the barriers that prevent the effective implementation of KM in organizations. Developed by Joshi, Parmer, and Chandrawat (2012), the KSBs model identifies and ranks the KSBs in the organization. Based on their model, there are ten KSBs including lack of culture, lack of ownership of the KM problem, lack of trust, lack of strategic issues, lack of motivation, lack of top management support, lack of methods and processes, resistance to change, and KM is not well understood. Among these KSBs, both the 'lack of top management commitment' and 'KM is not understood' are the driving barriers (Anand et al., 2013; Joshi et al., 2012).

In the context of SMEs, they have different attitudes towards KM rather than large corporations. Their idiosyncratic characteristics are embedded in the ownership and management,

structure, culture and behavior, systems, processes and procedures, human resources, and customer and market (Anand et al., 2013; Cerchione et al., 2016; Egbu et al., 2005; Yeow et al., 2013). These characteristics lead SMEs to be more likely to implement KM (Yew Wong & Aspinwall, 2004). For example, SMEs can take advantage of centralized decision-making to implement KM. Furthermore, SMEs have a simple structure, unified culture, people dominated systems and flexible processes, and direct communication. These advantages help small businesses to understand the importance of KM faster and enable them to implement KM in the entire organization with less resistance (Yew Wong & Aspinwall, 2004).

2.1.2 KM Processes

Scholars have realized that KM processes including knowledge creation, knowledge storage, knowledge transfer, and knowledge application (Alavi & Leidner, 2001) maximize the value of previously mentioned knowledge assets (Bhojaraju, 2005). Knowledge creation refers to developing new tacit and explicit knowledge within the organization through interactions among organizational members as well as collaborative activities (Alavi & Leidner, 2001; Nonaka, 1994). Knowledge storage also refers to organizational memory, and it is the process of preventing loss of the acquired knowledge. Organizations store knowledge through written documentation of organizational procedures (Alavi & Leidner, 2001; Nonaka & von Krogh, 2009). Knowledge transfer allows organizations to transfer and locate knowledge where it is needed through the communication processes and information flows (Alavi & Leidner, 2001). The application of knowledge refers to mechanisms that put knowledge in organizational routines. This way knowledge will continue to be applied and integrated in organization.

Developed by Nonaka (1994) SECI (Socialization, Externalization, Combination, Internalization) model is one of the widely cited theories in the knowledge management area. This model presents four modes of knowledge conversion to show how information becomes a piece of knowledge through a broad range of processes and the interaction between tacit and explicit knowledge. In this section, we will explain each mode by providing examples. Understanding the knowledge conversion steps is important as they could be considered as drivers for DevOps principles (Nielsen et al., 2017).

Socialization refers to the conversion of tacit knowledge to tacit knowledge (e.g., face-to-face interaction between individuals) (Nonaka, 1994). This can be done through mentoring

sessions in which one experienced person converts tacit knowledge to another person. Nonaka et al. (2000) argue that socialization requires a trustful environment and that it can occur by spending time together and living in the same environment. Externalization refers to the conversion of tacit knowledge to explicit knowledge (e.g., collective face-to-face interaction) (Nonaka, 1994). An example of externalization is when organizational team members gather to work on a single project together; they constantly discuss risks and technical issues to convey an individual's tacit knowledge into explicit knowledge through dialogue (Nonaka & von Krogh, 2009). Although each team member carries out one part of a task, they constantly team up and share their achievements to increase the visibility of the processes as well as awareness of the team about the project status.

Combination is the process of converting explicit knowledge into more systematic sets of explicit knowledge (e.g., collective and virtual interactions) (Nonaka, 1994). The main difference between combination and externalization is the way a group of people shares their knowledge. Here, virtual tools and technologies such as monitoring tools[3] are the key elements to convert the explicit knowledge into systematic and new sets of explicit knowledge (e.g., report and business plans). The converted knowledge then can be used by other stakeholders in the organization for further analysis (Nonaka, 1994; Nonaka & von Krogh, 2009).

Internalization refers to the process of embodying explicit knowledge into tacit knowledge (Nonaka, 1994). Using new technologies with subsequent group discussion is a suitable example here that lead to organizational learning. In this example, individuals assimilate knowledge obtained from other people through real practice and case studies (Nielsen et al., 2017; Nonaka, 1994; Nonaka et al., 2000).

Another view of KM process identified by Bhatt (2001) that categorized KM process into following phases:

- Knowledge creation: refers to the ability of the organization to develop novel and useful ideas and reconfigure and recombine existing pieces of knowledge through scanning opportunities, monitoring the external environment, and borrowing new technologies.

[3] Monitoring tools are used to continuously keep track of the status of the system in use, in order to have the earliest warning of failures, defects or problems and to improve them. There are monitoring tools for servers, networks, databases, security, performance, website and internet usage, and applications.

- Knowledge validation: is the process of continually monitoring, testing and refining the knowledge base to validate the knowledge, keep it up to date, and discard unnecessary knowledge.
- Knowledge presentation: refers to the way organizations display the information, data, and knowledge scattered in a different location to the organizational members. Such integration of knowledge presentation can be done by choosing similar standards, programming schemes, pre-defined templates in databases and different mediums.
- Knowledge distribution: refers to the interaction between organizational technologies, techniques, and people that can lead to knowledge distribution before it can be exploited. (e.g., intranet, e-mail, and bulletin board allow knowledge to be discussed, utilized, interpreted throughout the organization)
- Knowledge application: refers to the process of making knowledge more active and relevant for the firm in creating value. Here, the goal is to locate the right organizational knowledge in the right form and right place for example by repackaging current knowledge in a different context as well as training employees to be creative while using products, services, and processes.

These KM activities are necessary for organizations to be able to capitalize on knowledge and improve business performance and enhance competitive advantage.

2.1.3 Knowledge Management System (KMS)

Technologies can excel organizations by embedding knowledge into the above organizational processes (Alavi & Leidner, 2001; Cerchione et al., 2016; Davenport, 2010). Therefore, many organizations integrate and share knowledge through Knowledge Management System (KMS) which refers to a class of information systems applied to manage organizational knowledge (Alavi & Leidner, 2001, 1999; Corbin, Dunbar, & Zhu, 2007; Kubátová, 2013b). Information system literature has identified two models of KMS (Alavi & Leidner, 1999). The repository model of KMS (associated with the codification approach) refers to the storage of knowledge in knowledge bases (Alavi & Leidner, 1999). For instance, Electronic knowledge repositories facilitate knowledge reuse by coding and sharing best practices (Kankanhalli & Tan, 2004). The network model of KMS which is associated with the personalization approach focuses

on how to transfer knowledge by linking people (e.g., communities of practice[4]) (Kankanhalli & Tan, 2004). Drawing on these two models, Table 1, taken from Chua (2004), describes different types of knowledge storage and communication and their features.

Table 1: Types of knowledge storage and communication (Taken from Chua, 2004)

Types of communication services	Description
communication between users	Implemented through utilities such as file sharing and e-mailing
collaboration among users	Implemented through synchronous meeting and asynchronous discussion
workflow management	Allows user to manage workflow processes by supporting online execution and control of workflow
Types of knowledge repositories	**Description**
knowledge server	Consolidated huge amounts of data from multiple sources within an organization Builds content, creates references and establishes links among documents, Allows users to browse via a web browser, Organizes knowledge into administrator defined categories based on text index and meta data properties.

Common KMS technologies include intranets and extranets, search and retrieval tools, content management and collaboration tools, social networks, data warehousing and mining tools, and groupware and artificial intelligence tools (Chua, 2004; Kankanhalli & Tan, 2004; Marwick, 2001; Oseledchik et al., 2018). Table 2 summarizes the technologies that support the knowledge creation, sharing and reuse processes. Studies also identify Corporate Universities (CUs) as

[4] Communities of Practice (COP) are flexible groups of professionals who are interested in the similar topic and interact to discuss shared topics (Kankanhalli & Tan, 2004).

knowledge management tools that are established to provide specific training to employees (Scarso, 2017). Such an organizational system helps practitioners to (1) code and share the best practices, (2) create corporate knowledge directories, and (3) create knowledge networks (Alavi & Leidner, 1999). Moreover, it benefits organizations by enabling them to be flexible, innovative, fast enough against market changes (López-Nicolás & Meroño-Cerdán, 2011). Besides the above, modern technologies such as cloud computing, big data, social media, and e-commerce play a significant role in managing knowledge in enterprises (Cupiał, Szeląg-Sikora, Sikora, Rorat, & Niemiec, 2018). Likewise, in the context of SMEs, KMS has a positive impact on identifying the market opportunities, innovation, and operational management. SMEs are willing to share a variety of information through the KM platform such as information about funding opportunities, product, market opportunity research, market features, share purchasing, training, legal issues, use of specific business tools, administrative issues, quality management, etc. (Evangelista et al., 2010).

Table 2: Technologies that support KM processes

Technologies	Description
Social network analysis tools	Uncovers the pattern of knowledge flow within and across organizational boundaries.
Collaborative tools	Provides a platform to share knowledge whit one another. Key features include shared spaces, calendaring, workflow management services. May include peer polling feature to rank postings and enable selective reading.
Content management	Establishes a structure to create and maintain different types of content in text, image, and video format.
Concept mapping	Links several related concepts within a given theme or context. Allows the content to be categorized and indexed to ease future searches. Provides inter-disciplinary perspectives and facilitated cross-referencing.
Support knowledge creation through codification	Possesses capabilities to capture and codify knowledge held by experts.

2.1.4 KM Organizational Outcomes

Researchers and practitioners consider knowledge as the most important resource of the firm that can lead to competitive advantage and effective performance of the organization (López-Nicolás & Meroño-Cerdán, 2011; Nonaka, 1994; Obeidat et al., 2016; Zheng et al., 2010). Scholars underscored the role of KM in economic, market, technical, human, and organizational performance (López-Nicolás & Meroño-Cerdán, 2011; Obeidat et al., 2016; Zheng et al., 2010).

Moreover, KM processes (i.e., acquisition, conversion, and application) can also impact customer satisfaction as well as internal human-performance (Egbu et al., 2005; Gholami et al., 2013).

KM affects organizational performance through Organizational Learning (OL) positively (Liao, 2009). OL is the process of collecting, growing, and sharing an organization's body of knowledge. Nonaka and Takeuchi (Nonaka & Takeuchi, 1995) claim that such organizational learning capabilities are obtained through well self-driven learning by individuals. Given that OL is one of the goals of KM (King, Qureshi, Kamal, & Keen, 2009), organizations attempt to sustainably improve the individual and organizational knowledge by sharing knowledge and by implementing shared activities such as meetings, collaborative troubleshooting, communication tools and techniques. (Nonaka, 1994). Thus, one organization is unlikely to develop personal or group learning without KM.

KM is an antecedent of SMEs' financial and non-financial performance (Delen et al., 2013). For instance, SMEs' market-performance is affected by KM practices. Accordingly, the more the knowledge acquisition, knowledge storage, knowledge creation, knowledge sharing, and knowledge implementation, the more the customer satisfaction (Gholami et al., 2013). Zheng et al. (2010) argue that knowledge utilization is the most significant KM practice that directly impacts organizational performance. Furthermore, the effective implementation of KM can affect human-performance in SMEs positively. As a matter of fact, KM benefits human resource management in that it can offer better on-the-job training of employees and improve employee retention (Egbu et al., 2005). As such, KM leads SMEs to sustainable competitive advantages.

In addition to the above insights, ample empirical studies support the claim that strategic KM helps SMEs excel by enabling innovative activities. López-Nicolás and Meroño-Cerdán (2011) examine the consequences of KM on innovation among 360 Spanish firms. According to this survey, KM positively impact the performance indirectly through an increase in innovation capability. The same findings are achieved by a survey on 266 Jordanian consultancy firms showing that innovative activities heavily depend on two aforesaid KM components. Also, KM processes (i.e., knowledge acquisition, knowledge sharing, and knowledge utilization) positively affect innovation (Obeidat, Al-Suradi, Masa'deh, & Tarhini, 2016).

Besides the above organizational outcomes, KMS and technologies help an organization foster a flexible workplace in which organizations could be fast enough against market changes

(Alavi & Leidner, 1999; López-Nicolás & Meroño-Cerdán, 2011). For instance, agile entities take advantage of various collaborative tools to manage organizational knowledge and react to the changes quickly (Davis & Daniels, 2016; Smeds, Nybom, & Porres, 2015). Therefore, managing knowledge and using appropriate technologies lead organizations to enhance their speed and flexibility (Alavi & Leidner, 2001; Jarrahi & Sawyer, 2013).

Although the above examples are common, agile organizational entities such as DevOps teams expose different attitudes toward the KM to capture, store and share knowledge (Cram & Marabelli, 2018; Jarrahi & Sawyer, 2013). In the next section, we will also discuss this topic by comparing the nature of agile and traditional KM and by investigating agile KM practices and principles in more detail.

2.1.5 Traditional KM vs. Agile KM

Given that traditional KM processes are not enough for agile work environment (Cram & Marabelli, 2018), this section tries to enhance our understanding of agile KM practices by focusing on differences between traditional KM and agile KM in practice. We will provide tangible examples of how organizations can replace traditional KM practices with agile methods to share, create, store, and utilize knowledge in the agile environment such as DevOps teams.

The agile term was used in the software process context for the first time (Singh et al., 2014). Agile methodologies have grabbed great attention of software development researchers in recent years (Cram & Marabelli, 2018; Davis & Daniels, 2016) in that they benefit agile software development environment by "bringing programmers, testers and quality assurance employees together to ensure closer collaboration as a team as well as shorten the time between software releases from several months or years to weeks." (Nielsen et al., 2017, p. 2). Agile practices place a heavy emphasis on flexibility, collaboration, and quick change and development (Cao et al., 2009; Davis & Daniels, 2016).

From a practical viewpoint, agile refers to a socio-technical term that focuses on team-oriented development rather than rigid software development (Balijepally et al., 2006). Therefore, agile approaches help software development organizations go beyond the process-centered development approaches to more people-centered development approaches (Cram & Marabelli, 2018). Singh et al. (2014) suggest four agile values which are common in all agile practices:

individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, responding to change over following a plan. Likewise, Davenport (2010) believes that in contrast to the detailed engineering methods, agile methods are people-oriented, welcome change, and tend to adapt for a short span of time.

Early research has distinguished patterns of traditional and agile KM practices. For instance, knowledge documentation relies on extensive documentation in the traditional approach, however, in the agile approach it depends on "just enough" documentation such as story cards (i.e., short descriptions of desired system functionality) (Balijepally et al., 2006; Cram & Marabelli, 2018). Regarding the traditional knowledge repositories, explicit knowledge stored in formal ways such as documents, while agile knowledge repository is more informal and uses tools such as storyboards (Balijepally et al., 2006; Jabbari et al., 2016). Agile storyboard, also called a task board, is a graphic organizer that provides the holistic view of the project, and thus allows software development teams to quickly get a sense of standby tasks.

Balijepally et al. (2006) claim that although team composition in the traditional approach is role-based, the agile approach follows cross-functional practices with team members playing multiple roles within a given project. Regarding competence management, Cao et al. (2009) argue that formal status reports and direct managerial oversight play a significant role in the traditional approach, while in agile environments, the members can monitor the competency of their colleagues through collective ownership. Formal policies result in trust and care in the traditional approach, while collective code ownership, scrum, collaborative workspaces provide team members with a trustful environment (Cao et al., 2009). Scrum is an agile process beginning with a sprint[5] planning meeting and ending with a review meeting to discuss the issue for future use. Daily stand-up meeting is an example of Scrum that is often seen applied less formally in organizations (Davis & Daniels, 2016). Table 3 Presents terms that are used under the Agile umbrella in the literature (Atlassian, n.d.; Cao et al., 2009; Davis & Daniels, 2016).

[5] Sprint is at the very heart of scrum and agile methodologies. It is a set period of one week to one month, during which specific work has to be completed and made ready for review (Atlassian, n.d.).

Table 3: Foundational Agile Concepts

Agile Terminology	Definitions
Scrum	Scrum is a software development methodology that focuses on maximizing a development team's ability to quickly respond to changes. It describes a set of meetings, tools, and roles that work in concert to help teams' structure and manage their work.
Sprints	The time frame in which the work must be completed- often 30 days.
Standups	It is commonly known as Daily Scrum. Led by the scrum master, the team comes together for short daily meetings, in which they discuss what they have completed, what they are working on and any issues that are blocking the work.
Retrospectives	A retrospective is anytime the team reflects on the past to improve the future.
Backlogs	Is a prioritized list of work. A well-prioritized agile backlog not only makes release and iteration planning easier, it broadcasts all the things the team intends to spend time on.
Kanban	It is a popular framework used to implement agile software development. It requires real-time communication of capacity and full transparency of work. Work items are represented visually on a Kanban board, allowing team members to see the state of every piece of work at any time.

Unlike agile practices that suggest informal ways of training (e.g. pair programming and daily stand-up meetings), traditional KM practices follow the classroom style of using static training material in formal ways. Thus, there is continuous learning gained by practices such as postmortems (i.e., the open forum for determining and analyzing all aspects related to the project's lifecycle) (Jabbari et al., 2016; Lwakatare, Kuvaja, & Oivo, 2015), but typically only through end-of- project feedback sessions in traditional work contexts (Fruhling & De Vreede, 2006). In the agile context, that is where "debrief meeting" comes into play, this approach allows team members to self-correct, reflect upon recent experiences, discuss what went well and wrong over the entire lifecycle of an agile project (Jabbari et al., 2016). Davis & Daniels (2016) argue that such organizational learning is needed for competitive advantages in an agile environment. In the context of agile entities, organizational learning which is the process of collecting, growing, and sharing an organization's body of knowledge, focuses on blameless and learning culture rather than punishment culture (Davis & Daniels, 2016; Lwakatare, Kuvaja, & Oivo, 2016).

2.1.6 Conceptual Basis: Practice Perspective in KM

KM has been studied from a variety of perspectives (Vieru & Rivard, 2008) that conceptualize knowledge as core competitive advantage resource (Nonaka, 1994), organizational memory (Baltrusch, 2001), and organizational capability (Spender, 1996). In this study, we adopt the practice perspectives, which is highly recommended for organization studies in recent years (Cecez-Kecmanovic, Galliers, Henfridsson, Newell, & Vidgen, 2014; Mueller, Hutter, Fueller, & Matzler, 2011; Tavakoli & Schlagwein, 2016). The potential of practice perspective to contribute to our understanding of practices in general (Cook & Seely Brown, 1999) and in KM discipline, in particular, is well known (Mueller et al., 2011; Orlikowski, 2002; Tavakoli & Schlagwein, 2016). "Practices are central to any account of social phenomena relevant to social sciences, information systems, and organization studies" (Cecez-Kecmanovic et al., 2014, p. 815). In addition to the above insights, two reasons draw us to practice perspective as the conceptual foundation for this study.

First, the practice perspective provides a conceptual mechanism to identify the key components of KM practices in the context of this study. KM comprises different forms of knowledge as well as the process of knowing (Cook & Seely Brown, 1999), and the practice perspective promises to integrate these different aspects of KM (McIver, Lengnick-Hall, Lengnick-Hall, & Ramachandran, 2013). Knowledge in practice encompasses two epistemologies: *possession* and *knowing*. Possession conceptualizes knowledge as tacit and explicit knowledge that agents (individuals or collectives) have. *Knowing* is something that is part of the action and refers to something that organizations do, rather than have (Cook & Seely Brown, 1999; Tavakoli & Schlagwein, 2016). It refers to the timely, processual, and ongoing enactment of routine in a particular context by agents. Although routinized and repetitive, these actions allow creativity and invention (Tavakoli & Schlagwein, 2016). Knowing "represent an ongoing social accomplishment, constituted and reconstituted in everyday practice" (Orlikowski, 2002, p. 252), which is needed when individuals and groups use knowledge in interaction with others (Cook & Seely Brown, 1999). Therefore, "Practices are the actions engaged in by individuals, groups, units, and firms to accomplish the ongoing work of an organization" (McIver et al., 2013, p. 599). We use the term practice to identify the way in which work gets done and knowing how to do it (Mueller et al., 2011; Orlikowski, 2002; Tavakoli & Schlagwein, 2016).

The focus on tacit and explicit knowledge leads to identifying the unit that possesses knowledge which can be an individual or a group (Cook & Seely Brown, 1999). Individual knowledge exists in the minds of people who share knowledge with others. Individual knowing processes can be achieved through face-to-face interaction and learning-by-doing activities result in an increase of both tacit and explicit knowledge in an organization (Nonaka et al., 2000; Nonaka & Takeuchi, 1995; Orlikowski, 2002).

Group knowledge exists in the minds of a group of people with potentially diverse data and information (Nonaka & Takeuchi, 1995). Collective knowing processes can be achieved through activities such as common training and collective socialization which enhances the tacit knowledge of the organization, trust, commitments, the knowledge about procedures, and the knowing on how to innovate (Nonaka & Takeuchi, 1995; Orlikowski, 2002). The increase in these sharing activities then increases access to distributed knowledge (Mueller et al., 2011). By bringing together different individuals, we can build group knowledge and collective knowing which eventually facilitate the conversions of data and information to knowledge (Lakshman, 2007). Tacit knowledge can be converted into explicit knowledge with intensive interaction of individuals in the workgroup (Nonaka & Takeuchi, 1995).

The second reason for considering the practice perspective is that this view conceptualizes the way practices are enabled by the uses of multiple technologies (Tavakoli & Schlagwein, 2016). In practice perspective, it is recognized that "material objects participate in the accomplishment of practice, make the practice durable over time" (Hekkala, Stein, & Rossi, 2014, p. 3) and that the emergence of new technologies facilitate the knowledge and knowing processes (Davenport, 2010; Lakshman, 2007; Mueller et al., 2011). For instance, tacit knowledge can be brought to surface through virtual technologies such as collaborative tools and social networks (Kubátová, 2013a; Oseledchik et al., 2018).. Monitoring technologies enable individuals to monitor current activities and learn for future use (Mueller et al., 2011). This focus helps us to move away from seeing entities, people, and technologies characterized by boundaries that mutually impact each other, toward the performative nature of practices (Mueller et al., 2011; Tavakoli & Schlagwein, 2016). According to practice perspective thinking, people and technologies are enacted and reenacted in practice which allows them to continuously perform in a web of relations (Cecez-Kecmanovic et al., 2014; Orlikowski, 2002). Practices, in fact, put people and things in place as active elements

of practice (Cecez-Kecmanovic et al., 2014) to provide "a perspective on the entwinement of agent and material artifacts and how this shapes day-to-day practices of individuals" (Tavakoli & Schlagwein, 2016, p. 7).

In sum, putting the practice in place; individuals and groups along with the advanced elements of technological resources are key components of this view of KM that constitute the practice (s). This is consistent with the claim that KM has two main approaches: personalization (i.e., includes communication through a network of people and cross-functional teams) and codification (i.e., technological infrastructure for KM) (Hansen, Nohria, & Tierney, 1999). As such, according to the socio-technical literature on KM as well as practice perspective, KM practices can be divided into two main categories: people-centered practices and technology-centered practices. Figure 1, part of the initial conceptual framework of this study, shows how KM practice perspective promises to integrate these two aspects of KM.

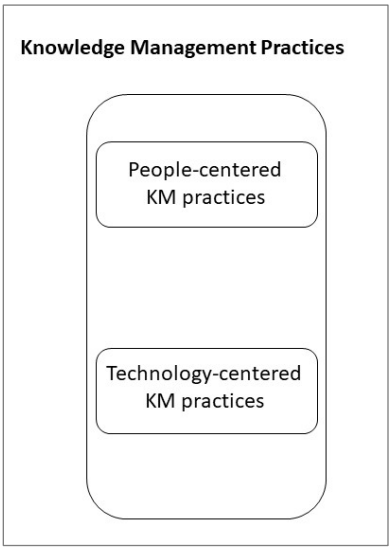


Figure 1: Practice Perspective in KM

For the purpose of this study, KM practice is conceptualized as a web of practices that are put in place for the purpose of identifying relevant data, conversion of these data and information to knowledge, managing both tacit and explicit knowledge, determining who knows what in order for knowledge to be used anywhere at any time (Alavi & Leidner, 2001; Lakshman, 2007; Nonaka et al., 2000). Investigating such a dynamic and continuous set of organizational practices and processual routines is important for exploring the DevOps behavior in the context of KM through

the grounded theory method. Adopting a practice perspective, we aim at developing a theory on the distinctive characteristics of KM practices that underpin DevOps teams.

2.2 DevOps

The review of DevOps literature was undertaken to identify the importance of DevOps, its definition, components, principle, and practices in the context of software development to enable the recognition of these factors represented by the DevOps teams in this study.

2.2.1 What is DevOps?

Traditionally, IT organizations considered clear boundaries, mindsets, skill sets, and culture for Developments (Dev) and Operations (Ops) teams (Davis & Daniels, 2016; Nicolau de França, Jeronimo, & Travassos, 2016; Sharma, 2017). The Dev view is to create innovation while the Ops is tasked with providing the users with a reliable and stable product. Traditionally, Dev and Ops lived in isolated worlds with limited scheduled interactions during release times.

After the advent of agile and the idea of continuous integration, developers and operations had to deal with more than one release daily (Sharma, 2017). This resulted in non-functional requirements during development on one hand, and on the other hand, caused unplanned programming bugs in operational systems (Tessem & Iden, 2008). Development teams aim to launch a new feature, any time, without hindrance, while the operations teams want to prevent interruptions and breaks in the system once it works. Therefore, the teams' goals are in tension in nature in the agile context (Davis & Daniels, 2016).

Moreover, practitioners believe that the split between these two groups in software development companies gives rise to direct and indirect costs as the size of the team, the service, and its traffic grow (Murphy, Beyer, Jones, & Petoff, 2016). They believe that the indirect costs come from the fact that these two groups are different in using vocabulary to describe situations as well as in understanding the possible risks and target level of product stability. This can lead them to use quite different approaches to address their goals, trust, communications, and respect which results in often more expensive costs for the organization. At their core, the ops teams focus on product visibility over how quickly it can be released and thus have more "launch review[6]" that

[6] Launch Readiness Review (LRR) is a self-reporting checklist the product teams conduct before the service goes live to customers.

could be a long list. At their core, the dev teams attempt to launch new feature requested by users quickly and thus have fewer "launches" and more "incremental updates[7]". The more the launch reviews for dev teams, the more the downtime of the software product and the more cost. When these two teams come together through understanding each other's concerns, they adopt tactics to split the product and expose the product to fewer launch reviews, and thus save costs on maintenance and upgrades (Kim et al., 2016; Murphy et al., 2016).

The DevOps paradigm, thus, promises to overcome these traditional boundaries between development and operation teams and to improve the common culture and shared goals (Davis & Daniels, 2016; Erich et al., 2014b; Nielsen et al., 2017). DevOps, a portmanteau of Development and Operations, is a cultural movement towards a way of thinking and working that encourages individuals and organizations to adopt practices such as sharing stories and developing empathy (Davis & Daniels, 2016). This modern software development method is a response to interdependency challenges by unifying tools and techniques that enable communications, collaborations, and efficient teamworking (Guerriero, Ciavotta, Gibilisco, & Ardagna, 2015). DevOps attempts to excel software development by having developers work closely together with operations. This convergence between development and operations helps organizations to increase trust, transparency and shared understanding among teams (Davis & Daniels, 2016; Erich, 2019). This solution enables a "balance between innovation and stability and between the speed of delivery and quality." (Sharma, 2017, p. 8).

Erich et al. (2014b, p. 8) define DevOps as "*a development methodology aimed at bridging the gap between Development and Operations, emphasizing communication and collaboration, continuous integration, quality assurance and delivery with automated deployment utilizing a set of development practices*". As Nicolau de França (2016) explain, DevOps is frequently associated with the idea of a movement of Information Technology (IT) practitioners, and they define DevOps as "*a neologism representing a movement of ICT professionals addressing a different attitude regarding software delivery through the collaboration between software systems development and operation functions, based on a set of principles and practices, such as culture, automation, measurement and sharing.*" Acknowledging all DevOps definitions, Erich (2019) argues that "at

[7] An incremental update contains only the change or difference from the previous update. The incremental build model is a method of software development where the product is designed, implemented and tested incrementally (a little more is added each time) until the product is finished.

the core of each definition of DevOps will be the interaction between development and operations".

2.2.2 DevOps Components

This section will discuss DevOps components which frequently mentioned by extant DevOps literature. We will explore DevOps components in more detail in that they are the key concepts in the practice perspective of this study.

2.2.2.1 DevOps Team

"*Team is a group of people working toward a common goal, interdependent with each other, with some familiarity among members.*" (Davis & Daniels, 2016, p. 118). DevOps team is typically formed from development and operations personnel (multiple DevOps engineers) interacting with each other and treating as a member of a single team (Erich, 2019).

Site Reliability Engineering (SRE) team is another term developed by Ben Treynor, who founded Google's Site Reliability Team. Literature frequently used SRE and "equivalently view SRE as a specific implementation of DevOps with some idiosyncratic extensions." (Murphy et al., 2016, p. 5). The underlying theme for both DevOps and SRE is to help involve IT function in each phase of a system's design and automate the operations (Washington, 2016). For this study, DevOps and SRE, regardless of their definitions, share many common principles; SRE's principles are consistent with many of DevOps core principles and practices (Murphy et al., 2016; Washington, 2016) that we will discuss later in this study.

2.2.2.2 Cross-Pollination of Skills

"*A skill is a combination of ability, knowledge, and experience that enables a person to do something well.*" (Hemon, Lyonnet, Rowe, & Fitzgerald, 2019, p. 3). Literature divides skills into hard and soft skills. Hard or technical skills refer to knowledge of programming language, scripting, testing, troubleshooting, operating system, etc. Soft or behavioral skills refer to those that are facilitating the communication, inspiring the team, supporting experimentation, etc. (Bang et al., 2013; Sharma, 2017). Scholars claim that both are necessary for a successful outcome in organizations (Hemon et al., 2019; Smeds et al., 2015).

Regarding technical skills, according to Google, SRE teams end up with a team of people who prefer not to perform the task by hand and have the skill set to write required complicated

software to replace manual work. Accordingly, hiring engineers with software expertise would help as they not only put 50% of their time doing operations tasks but also, they can put their remaining time for Dev tasks as they inherently have the ability to design and automate the system (Murphy et al., 2016). Based on literature, DevOps teams must involve people who have broad knowledge in front-end and back-end to be able to have the same level of understanding of problems and to solve them together (Bang et al., 2013; Davis & Daniels, 2016).

2.2.2.3 Cross-Functional Tools

In order to have effective DevOps, it is important to select and use effective and upgraded tools that facilitate troubleshooting (Davis & Daniels, 2016; Smeds et al., 2015). According to Erich (2019), DevOps term can be applied to DevOps tools in different ways. DevOps tools could refer to those tools that foster interaction between development and operations (e.g., Jira). Other than that, DevOps tools defined as a means that could bridge the disciplines of development and operations (e.g., Jenkins). Moreover, DevOps tools could be defined as tools that automate the interaction between development and operation personnel (e.g., Ansible and cloud engine) (Erich, 2019).

2.2.2.4 DevOps Software Delivery Stages

For the purpose of this study, it is important to have a holistic understanding of software delivery processes in that several common practices performed by both software development and operations teams under these stages (Nicolau de França et al., 2016).

In the context of software delivery, there are four stages: *plan & measure*, *develop & test*, *release & deploy*, as well as *monitor & optimize* (Nielsen et al., 2017). The *plan & measure* stage refers to the focus of both Dev and Ops teams on the requirements and needs of users. IT operations must be involved early in this stage to address the risks associated with the business plan and design decisions (Kim et al., 2016; Nielsen et al., 2017). In *develop & test* stage, Dev and Ops team must carry out the quality assurance through continuous and automated testing and integration. Tools such as Git or Subversion go hand in hand with the software delivery in this stage in order to ensure documentation and tracking of codes, and Puppet can be used for configuration management to show the desired status (Nielsen et al., 2017; Sharma, 2017).

The *Release & deploy* stage refers to the frequent releases of smaller software packages as well as the continues deployment of small changes as soon as they are released (Nielsen et al., 2017). This helps developers to recognize the impact of their changes and reduce the risk of failures (Lwakatare et al., 2016; Sharma, 2017). Tools such as Jenkins helps DevOps team to automate the non-human part of the software development process and focus on selecting software versions that are ready for release (Sharma, 2017). The *Monitor & optimize* stage provides relevant data to stakeholders through the continuous monitoring based on measurement metrics such as cycle time, meantime to detect and repair, as well as user feedback. Here, continuous improvement and feedback are the main goals so Ops team must give feedback to Dev team about the health of the system in production (Davis & Daniels, 2016; Erich et al., 2014a; Nielsen et al., 2017). Figure 2 summarizes each software delivery stage with related practices.

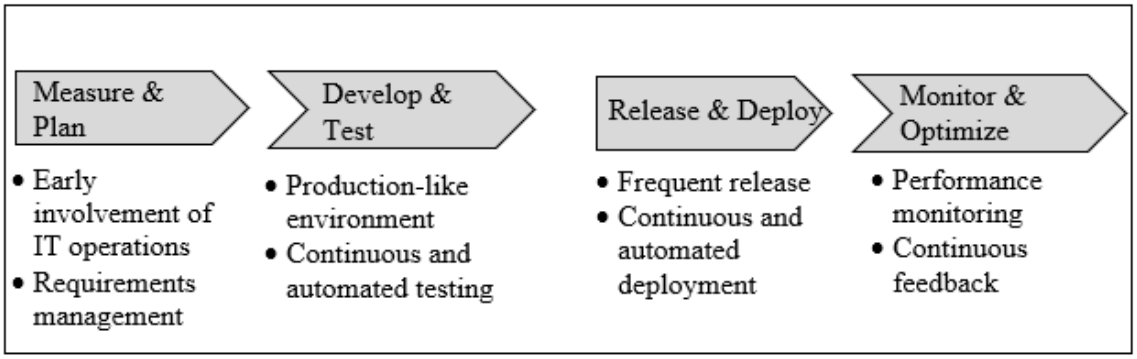


Figure 2: DevOps Stages and Practices (Nielsen et al, 2017)

Based on our literature review (Kim et al., 2016; Nielsen et al., 2017; Sharma, 2017), Table 4 attempts to present some examples of DevOps tools that are used in various stages of software delivery. In addition to listing common tools in the industry, we also provide a description of one popular tool in each category.

Table 4: DevOps Software Delivery Stages and Tools

DevOps Software Delivery Stages	DevOps Tools	Description
Measure & Plan	Jira, Redmine,Trac,Rally, splunk	JIRA is an Agile-friendly planning tool that supports DevOps workflows that require continuous feedback loops, collaboration with multiple project teams, and open communication.

Develop & Test	DevOps CI/CD[8] tools: Jenkins, Bamboo, Apache ActiveMQ, Codeship, Snap CI. Version control and configuration management tools: Ansible, AWS CloudFormation, Git, Subversion, Terraform Testing Tools: Appium, Pytest	Jenkins is an open source automation server that provides a plugin architecture to support continuous integration and delivery. It integrates with a variety of software tools in the CI/CD toolchain and distributes work across multiple platforms.
Release & Deploy	Release management tools: Serena, DBmaestro, XL release Deployment tools: Capistrano, AWS codeDeploy, Go, Deploybot	Capistrano is a remote server automation and deployment tool primarily used for deploying web applications.
Monitor & Optimise	Monitoring tools: Elasticsearch, Grafana, Splunk, Kibana, New Relic. Log management tools: Sentry, Logsene, Loggy Process supervisor tools: Monit, God, runit	New Relic is SaaS web and mobile application performance monitoring tool that provides analytics from the customer experience perspective. It monitors availability, alerting, and notifications in real time for applications running in cloud, on-premise, or hybrid environments.

Considering that DevOps is defined as a set of practices and principles (Nicolau de França et al., 2016; Sharma, 2017), it would be worthwhile to take a deeper look at the definitions of practices and principles. Aiming a better understanding, the next section will investigate extant literature on DevOps' principles and practices.

2.2.3 DevOps Principles and Practices

Since this study attempts to investigate KM efforts toward the improvement of DevOps principles and practices, this section first explores the widely used categorizations of principles and then goes deep through their definitions and related practices that materialize them. Moreover, the relation between DevOps principles and organizational outcomes will be discussed. This section investigates those principles and practices which are fitted with the purpose of this study and which are stated frequently by interviewees.

Studies show that there is a close relation between DevOps principles and practices (Nicolau de França et al., 2016; Nielsen et al., 2017; Sharma, 2017) and that "the practices materialize the

[8] CI/CD generally refers to the combined practices of continuous integration and either continuous delivery or continuous deployment. In the context of corporate communication, CI/CD can also refer to the overall process of corporate identity and corporate design.

principles into daily development and operations activities." (Nicolau de França et al., 2016, p. 57). The literature identifies different categorizations for DevOps principles. Proposed by Humble & Molesky (2011) CAMS: *Culture*, *Automation*, *Measurement,* and *Sharing* are the main principles of DevOps. In another attempt, Kim, Humble, Debois, & John (2016) claim that DevOps principles refers to the *principle of feedback* and the *principle of continual learning*. "These are the principles that enable the constant creation of individual knowledge, which is then turned into a team and organizational knowledge." (Kim et al., 2016, p. 37).

Davis & Daniels (2016) characterized DevOps by common pillars which are vital for the true way of doing DevOps including *Collaboration, Affinity, Tools*. In another viewpoint, DevOps is characterized by values such as *Agile*, *Collaborative*, and *Integration* (Nielsen et al., 2017). Considering the overlapping principles, at the following we will look at how DevOps could fit with these categorizations.

2.2.3.1 Culture of Collaboration and Affinity

Highly recommended by literature, a *culture of collaboration* between the software development entity and the operations entity is one of the main principles in DevOps teams (Erich, 2019; Erich et al., 2014b; Jabbari et al., 2016; Lwakatare et al., 2015; Nicolau de França et al., 2016). Such culture is "the one that encourages empathy, support and a good working environment for those involved in software development and delivery processes" (Lwakatare et al., 2016, p. 96). DevOps teams need to empathize with each other and also with software users to help each other understand the needs, push code quickly, and deliver the best software (Davis & Daniels, 2016; Erich, 2019).

This principle can be seen through rethinking and reorientation of roles and teams in DevOps activities that can be done by developers that set up shared workplaces with operations to pay closer attention to deployment script, attend their stand-ups, conduct troubleshoot together, and solve a problem as one team (Lwakatare et al., 2016). According to Nielsen (2017) collaboration refers to a culture that enables development and operation teams to interact regularly through weekly meetings, less formal communications, shared responsibilities, role rotation, etc. Culture of collaboration and shared responsibilities is a must for delivering new capabilities without conflicting interest and for delivering a high-quality product (Lwakatare et al., 2015; Nielsen et

al., 2017; Smeds et al., 2015). Therefore, the culture of collaboration as a social aspect of principles is based on respect, trust and open communication atmosphere (Nicolau de França et al., 2016).

Interestingly, Nicolau de França (2016) argues that most of the DevOps practices belong to collaborative practices that aim to foster collaboration among team members such as role rotation, shared responsibility, face-to-face meetings, integrated development, and test, etc. These recurrent common practices allow both sides to get involved in all stages earlier, understand the common problems, exchange skills, and experiences, and eventually improve the status quo (Davis & Daniels, 2016; Erich et al., 2014b; Jabbari et al., 2016).

While the *collaborative* principle emphasizes relationships between individuals, the *affinity* principle focuses on strong relationships between teams and departments within the organization (Davis & Daniels, 2016). These two inter-team and intra-team principles together can foster empathy, a collective body of technical and cultural knowledge, and learning among groups of people (Davis & Daniels, 2016; Lwakatare et al., 2016).

2.2.3.2 Agile

Scholars agree that DevOps builds on *agile* principles and methodologies (Davis & Daniels, 2016; Erich et al., 2014a; Nielsen et al., 2017). Agile methods are a people-oriented, welcome change, and tend to adapt for a short span of time (Davenport, 2010). DevOps has grown around agile methodologies that are putting more values on "individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, responding to change over following a plan" (Davis & Daniels, 2016, p. 33). DevOps often use agile methods such as daily Scrum, Kanban, Sprint planning, etc. (Davis & Daniels, 2016; Smeds et al., 2015).

Although DevOps practices and agile practices share many similar characteristics, there are some idiosyncratic practices associated with DevOps (Davis & Daniels, 2016; Jabbari et al., 2016). For instance, both DevOps and agile practices require continuous integration and testing. However, DevOps further advance practice in organizations by continuously monitoring the performance of the system and by presenting results in automated dashboards (Jabbari et al., 2016).

2.2.3.3 Automation and Integration

Automation is one of the core principles that characterizing most of the practices that constitute DevOps (Murphy et al., 2016; Tessem & Iden, 2008). Instead of manual deployment, automation of infrastructure and operations processes is needed in order to increase flexibility, repeatability, and delivery speed in agile software development environment (Lwakatare et al., 2016). DevOps automation is supported by practices such as cloud-based e-mail delivery service, cloud-based logging service, and real-time user monitoring tools (Erich et al., 2014b). This is consistent with the claim that agile practices bring greater speed, flexibility, and real-time backup capabilities (Cram & Marabelli, 2018).

The *Integration* of practices and tools, itself, which is achieved through automation, is considered as DevOps principle in some literature (Nielsen et al., 2017). Accordingly, seamless integration is achieved through use of tools, best practices, and automation of tasks; the increasing number of tools lead organizations to integrate them using cloud services. Cloud environment together with automation enables the infrastructure to be provisioned and functions to be reliable and repeatable (Lwakatare et al., 2015).

2.2.3.4 Feedback

DevOps is characterized by a *principle of feedback* (Kim et al., 2016). By addressing this principle, Dev and Ops team detect the problems when they are cheaper, smaller, and easier to solve. This can be done through continuous monitoring that runs from development to operations cycle (Nicolau de França et al., 2016; Sharma, 2017). Ops teams must provide feedback to Dev team about the system performance, health of the product, user's satisfaction through continuous monitoring of processes and users (Lwakatare et al., 2016; Sharma, 2017). This allows teams to be informed about future problems during earlier stages and thus enhances the safety and health of the system of work in the long term. (Kim et al., 2016). Importantly, the feedback principle lets people determine what to change and how to act, so it benefits both individual and organizational learning (Davis & Daniels, 2016).

2.2.3.5 Continual Learning

Due to the benefit it could promote, *continual learning* is considered as one of the main principles of the DevOps movement (Kim et al., 2016; Smeds et al., 2015). This principle is about

creating a continual and dynamic system of learning. Here developers and operations are lifelong learners who learn from successes and failures in a generative learning system[9] (Davis & Daniels, 2016). In contrast to bureaucratic organizations in which new ideas create problems, generative entities (here DevOps teams) establish a dynamic system of learning by sharing the responsibilities, welcoming the new ideas, and rewarding the bridging between teams (Davis & Daniels, 2016; Kim et al., 2016).

This can increase both individual and organizational learning through practices such as searchable and shared documents, internal discussion groups, and widely blameless postmortem reports (Davis & Daniels, 2016; Nicolau de França et al., 2016). A remarkable example here is that in postmortem meeting that occurs when the events' outcome was surprising or unplanned, team members discuss the unexpected happenings, share their viewpoints, and investigates the incident, and thus grow the knowledge. Moreover, this principle helps DevOps team to take risk, and integrate improvement and learnings into their daily works to be able to solve problems once occur (Kim et al., 2016; Sharma, 2017).

2.2.3.6 Sharing

According to this principle, information and knowledge should be spread to Dev and Ops personnel (Nicolau de França et al., 2016; Tessem & Iden, 2008). "DevOps is about development and operation personnel sharing about their involvement in different parts of the same system." (Erich, 2019, p. 98). Sharing is crystalized through practices such as placing Dev and Ops personnel in the same team, making documentation understandable for both sides, hiring employees with skillsets of both development and operations (Erich et al., 2014a; Jabbari et al., 2016). Disseminating knowledge among individuals increases their awareness about the changes and new characteristics of products, promote the workflow visibility, and eventually improve the team performance (Nicolau de França et al., 2016).

Relying on above DevOps principles, IBM identifies patterns of DevOps practices and capabilities that make up DevOps including think, code, deliver, run, manage, learn, and culture. In another attempt, Sharma (2017) claims that at the core of all these DevOps practices *continuous*

[9] Generative Learning means that new ideas must be integrated with pre-existing personal experience, previously acquired knowledge, and learner cognitions. (Wittrock, 1974)

integration and *continuous delivery* are two key capabilities of DevOps which are focusing on minimizing the cycle time.

In sum, building on agile practices, DevOps is the collection of principles and practices that try to improve cooperation between software development and technology operation in organizations (Smeds et al., 2015). However, the DevOps approach aims to take the agile approach one step further by using IT in an integrated environment of operations, development and maintenance (Nielsen et.al 2017). DevOps practices are being heralded as an effective approach to managing technology change and delivering technology solutions in a highly competitive business environment (Jabbari et al., 2016; Nicolau de França et al., 2016). Since the software is created and is used by people who are involved in such cultural movement, DevOps aims to address the integration of social structure, culture, and technology in order to work more effectively toward shared goals (Davis & Daniels, 2016; Nielsen et al., 2017).

Following the above concepts of agile methodologies and practices, and after understanding both DevOps and KM components, the next section will show to what extent the prior literature has considered KM in the context of DevOps.

2.3 KM and DevOps

Previous research identifies the knowledge sharing topics in development and operation communities. For instance, building on SECI model, (Nielsen et al., 2017) present the required knowledge conversion between Dev and Ops teams. Table 5, summarizes four modes of knowledge sharing conversion with related DevOps activities.

Table 5: knowledge-sharing modes in DevOps (Nielsen et al., 2017)

Knowledge conversion modes	DevOps activities
Socialization Individuals' face-to-face interaction Tacit to Tacit	• Face-to-face meetings • Pair programming • Developer rotation
Externalization Collective face-to-face interaction Tacit to Explicit	• on-site customer representatives • meetings between IT Development and IT Operations
Combination Collective & virtual interactions Explicit to Explicit	• Using shared log files & joint platform
Internalization Learning by doing Explicit to Tacit	• New technology with case-study.

These four modes can support DevOps values such as collaboration, integration, and agile environment (Nielsen et al., 2017).

Bang et al. (2013) focus on an individual's knowledge and argue that individual's Knowledge, Skill, and Abilities (KSAs) support the four perspectives of DevOps (i.e., collaboration, automation, measurement, and sharing). Another, though different DevOps KM framework is suggested by Wettinger, Andrikopoulos, & Leymann (2015), focusing specifically on the technical aspect of DevOps (e.g., Cloud computing, servers, database, and APIs[10]). It shows how DevOps knowledge can be captured in knowledgebase and be used automatically for cloud application.

In the same sense, Corbin et al. (2007) attempt to couple the development and operation communities by developing a three-tier knowledge management scheme. They explain how the following knowledge exploration, evaluation, and execution practices help software engineering organizations to continually observe the technological trends, plan for innovation, provide a feasibility report, test and collect both teams' feedback.

[10] application programming interface (API) is a computing interface to a software component or a system, that defines how other components or systems can use it.

- Exploration refers to continuous observation of technological trends, development plans for software innovation and improvement.
- Evaluation is tasked to provide a feasibility report and specification of software product.
- Execution is tasked to develop the software prototype, test and collect feedback, and conduct troubleshooting.

Drawing upon these previous efforts on describing the interrelationships between KM and DevOps, this study aims to contribute to the literature by developing a practice perspective of the role of KM in DevOps, and by tying specific KM practices to positive DevOps performance outcomes. In the next section, the initial framework derived from literature is presented showing how a practice perspective of KM discipline might usefully be leveraged to advance DevOps research.

2.4 Initial Conceptual Framework

The summary of the above-mentioned literature on DevOps and KM is presented in Figure 3, the initial conceptual framework of this study, that shows two main components of our research question and their posited interrelationships: KM practices and DevOps practices and performance. This initial framework is conceptualized to help answer the main research questions in this study: 1) What are the enablers of KM practices in DevOps teams? 2) What are the distinctive characteristics of KM practices that underpin positive DevOps performance outcomes?

More specifically, Figure 3 that is derived from the literature and the practice perspective of this study attempts to show how KM practices could be drivers for DevOps performance and principles. It is important to note that this high-level framework aimed at guiding the rest of the study in terms of data collection and data analysis (Yin, 2017). Therefore, the refined version of this framework will be presented in the result section providing more details in each part of the initial framework as well as new emerging aspects based on the findings of this study.

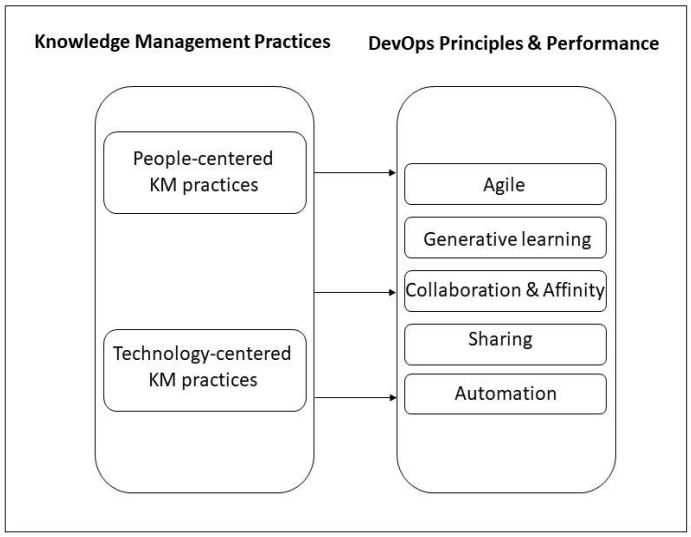


Figure 3: Initial Conceptual Framework

3 Research Design and Methods

3.1 Research Setting and Design

Using an inductive research design and qualitative research methods, this study follows a multiple case study approach to develop a theoretical perspective on the role of KM practices in DevOps. To address the research questions, five main steps were carried out (as shown in Figure 4). First, we looked at extant literature on KM and DevOps. This leads to developing an initial conceptual framework reflecting the interrelationships between KM and DevOps principles and performance. The exploratory nature of the conceptual framework allows addressing the research question by initiating data collection and inductive coding (Yin, 2017). As such it highlights how KM practices can be drivers for DevOps performance. The framework is meant to be revised and enhanced after conducting the data collection and analysis and presenting the results that were unknown before this study.

After developing an understanding of key components of research questions (e.g., DevOps practices, KM practices and technologies), we conducted a pilot project whereby we interviewed two key personnel at a company with DevOps. Out of these, one was a senior DevOps engineer and the other interview was conducted with a professional service expert who had close interaction with DevOps teams. The interviews were recorded and coded manually on the printed data source. In addition to the inductive codes (emerged from data), we also explored deductive codes from the existing literature in the context of DevOps KM. These interviews were important to update the initial framework, validate our research questions, test and refine interview questions.

The findings of the pilot project helped to better understand DevOps tools and technologies, the current state of DevOps in various sectors, and commonly used techniques for knowledge conversion between Dev and Ops teams. The insights from the pilot project were used to further explore DevOps practices by which tacit and explicit knowledge can be managed. The findings also helped in refining data collection procedures int terms of participant selection and appropriate interview questions to be asked. Given that professional service expert also provided us with valuable information on how they use collaborative technologies to work on common projects with DevOps teams, we decided to add non-DevOps middle-management personnel besides DevOps

personnel in our data collection plans. This allowed for a better understanding of inter-team and intra-team knowledge sharing practices.

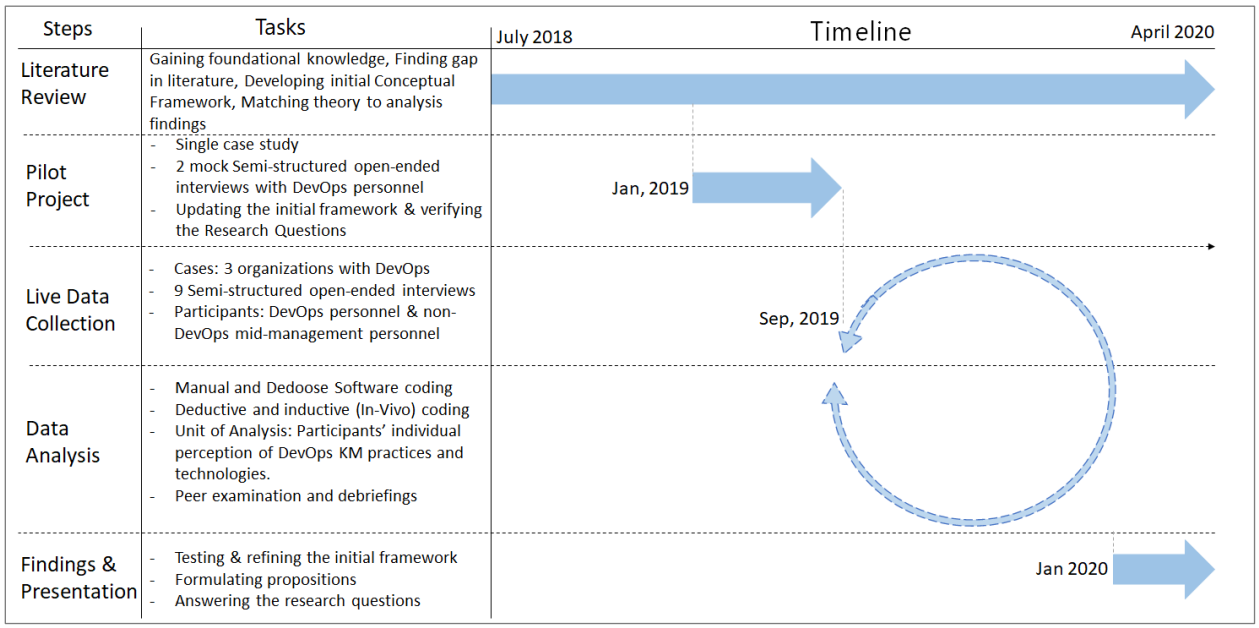


Figure 4: Research Plan and Activities

This study follows a multiple case study approach which is an appropriate method to answer "how" questions, and thus, can answer our research question on how KM practices could be the drivers for DevOps principles and accordingly for business impacts and vice versa (Eisenhardt, 1989). To address the research question, we employ a multiple case study approach to compare and contrast different contexts (Eisenhardt, 1989) – in this case, different organizations with varying degrees of DevOps scale and maturity. Compared with single case studies as we conducted in our pilot project, multiple case studies of 3 organizations with DevOps teams allowed for more generalization and extension of initial framework (Eisenhardt, 1989; Miles, Huberman, & Saldaña, 1994).

It should be noted that the progress arrows in Figure 4 merely show the approximate starting point of each step after another. After conducting the literature review on KM and DevOps, we started to conduct a pilot project. Upon finishing the pilot project and receiving insightful feedback, we validated our research design to collect the data through multiple case studies. After the first 2 or 3 interviews, we started the data analysis and coding approaches.

Given that live data collection and data analysis steps need to be done continuously and in parallel, we bolded these two steps by drawing a cycle around to show how we might proceed the cyclic process of data collection and data analysis simultaneously. Finally, the findings and discussion step happened after finishing the data analysis. As Figure 4 shows, our literature review spans across all phases of the research. Initially, the literature review has helped with the conceptualization of our initial framework, and in later stages, the literature review will help with theory matching to validate our research findings.

3.2 Data Collection

To improve our access to the data of KM practices of DevOps teams, we collected data through semi-structured open-ended interviews lasting approximately 30 to 40 minutes. According to Creswell and Poth (2018, p. 36) "the more open-ended the questioning, the better, as the researcher listens carefully to what people say". Open-ended questions allowed the study participants to provide the interviewer with more examples and experiences.

Our semi-structured interview consists of predefined themes and sub-questions. This approach is useful to keep the interviewees on track. Patton (2015) states that such an interview guide helps ensure that the interview is rich in data. Conducting a preliminary study in the subject area of DevOps Knowledge Management Practices, we conducted interviews with 9 participants in 3 organizations for our study in the following roles: DevOps Development Personnel, DevOps Operations Personnel, and other pertinent Non-DevOps Middle-Management Personnel. In general, we asked participants to talk about: (1) evolution of DevOps over time in the organization; (2) the connection between knowledge management practices and DevOps practices; and (3) the role of technology in KM in DevOps teams. Appendix 7.2 outlines our interview open-ended general questions as well as probing questions that were used to obtain more information in specific area(s).

In order to gain a better understanding of knowledge management (KM) practices in DevOps teams, we restricted our sample to firms with DevOps teams and participants with relevant job titles. Potential survey respondents were recruited through a combination of convenience and purposive sampling techniques. Applying these criteria, we polled employees and collected data from 3 organizations that had verbally agreed in principle to help with this research. Moreover, additional participants were sought through a snowballing technique where participants

recommended other colleagues who may be contacted for help with this study. We provided each participant with our Call for Participation information and a Consent form. Participants were asked to indicate their agreement with the procedures followed in this research study. (see Appendix 7.3 and 7.4)

We provided paper-based forms for in-person participants. The electronic version of the forms was made available for phone call interviews. We provided an online link to the consent form at the beginning of the phone interview, and the participant needed to indicate consent before we started the interview. The interviews were voice recorded on multiple audio recording devices with prior consent of the participants. Then the records were transcribed to text by the principal researcher. Detailed notes were also taken during the interviews in order to modify or add new probing questions for upcoming interviews. In addition to the transcribed interview data, we also reviewed papers, reports, and documents suggested by the interviewees. These documents helped us to verify the informants' answers and provided further insights for analysis as well as for future interviews.

3.3 Data Analysis

First, we reviewed the literature with the goal of understanding the definitions and components of KM and DevOps in the extant literature as well as for identifying specific concepts pertinent to this research area. This lead us to develop an initial conceptual framework guiding the data collection. According to the literature review there is a strong call for more qualitative approaches including grounded theory (GT) in exploring the role of KM in DevOps (Nielsen et al., 2017). Since the role of KM practices as enablers of DevOps' performance has not adequately been addressed in previous literature, a grounded theory coding procedure can potentially help to build a theory on how KM practices drive DevOps principles and subsequently organizational performance.

Grounded theory coding procedure required analysis of data inductively through a theoretical sampling and comparative analysis (Charmaz, 2006; Hennink, Hutter, & Bailey, 2011). In terms of theoretical sampling, this study uses DevOps personnel as sample choice which is consistent with the area of the study. 9 in-depth interviews with participants from 3 organizations were used in this study. We used open-ended coding wherein we analyzed each data that were recorded and transcribed. The transcribed data were analyzed paragraph by paragraph using the

Dedoose software to identify potentially relevant factors to the key components of research questions as KM practices and DevOps performance (e.g., explicit knowledge, tacit knowledge, KM barriers, DevOps values, knowledge sharing technologies, etc.). In addition to prespecified codes based on literature review, we also used emergent content analysis to generate inductive codes. Inductive codes that emerged from the data and were deemed relevant to knowledge management practices in DevOps teams helped us stay close to the interviewee's statements. Such emergent content analysis and inductive coding practices have been recommended by various researchers (Charmaz, 2006; Hennink et al., 2011).

The analysis of this study proceeded in three steps. Using the data structure developed by Gioia, Corley, and Hamilton (2012), we summarized our analysis in three steps (presented in results section, Figure 5). This data structure shows how we chose primary codes and secondary codes. It also depicts how we progressed from secondary codes to aggregate dimensions. "The Gioia approach's uniqueness is in its process of moving back and forth between the emergent findings and the literature/theories to identify, generate or position new concepts in or from the data collected." (Chandra & Shang, 2019, p. 12).

In the first step of data structure approach, staying close to what the participants mentioned, we obtained a large number of primary codes. Then, we collapsed the repetitive primary codes to obtain a final set of primary codes. For instance, participants frequently mentioned how they suffer from lack of documentation practices in their team. In DashCo.P02's words:

> "lately we had layoff, and somebody left the team and their knowledge left the team… Having known we've gone through this were made my effort to increase the coverage of documentation of what we had in Confluence".

According to DashCo.P05:

> "Documentation is really hard to keep up to date. And when everything is changing all the time, you can spend a lot of time updating your docs"

To address such repetitive concerns, we called them as a first-order code "*awareness of negative consequences of weak documentation practices*".

In the second step, we discussed the first-order codes to interpret them. To unify the codes, we constantly moved between the interviewees' statements and prior literature. This helped us to develop themes from the abstracted data and obtain the second-order codes. For instance, the first-order codes such as "*awareness of negative consequences of weak documentation practices*" and "*awareness of negative consequences of weak communication*" were summarized into second-order code as "*challenges*". Moreover, in order to refine the emerging theme, a literature review was used.

To develop codes around DevOps performance, we wrote down keywords for different aspects of performance identified by DevOps literature (e.g., organizational learning (OL)) and then we searched for their possible synonyms (e.g., teach, memorize, find out) and highlighted paragraphs in which DevOps personnel talked about these topics with their own words (e.g., self-driven learning). In doing so, we capture first-order codes around three main DevOps performance theme: *individual/organizational learning, generating new ideas, quick reaction and proper decision making,* and *health of the system.* We developed these themes when interviewees talked about the impact of KM practices in their daily works.

In the third step of data structure approach, the second-order codes lead us to three main aggregate dimensions that constitute the main components of research question and final model (DevOps perception of KM importance, DevOps KM, DevOps performance). We evaluated each case individually by focusing on how KM practices and technologies are drivers for DevOps performance. The comparison and evaluation extend across the 9 DevOps personnel to identify the patterns and stopped when there was theoretical saturation meaning additional data collection did not improve the emerging relationship that had been identified (Hennink et al., 2011). The possible relationships between our codes within and across cases and as well as extant research were discussed which led us to develop the propositions constituting the GT and final model (Eisenhardt, 1989).

Some codes were extracted exactly based on the terms and statements of the interviewees. Such an approach called in-vivo coding, which is useful in highlighting the statements of participants and giving meaning to the data (Hennink et al., 2011). For instance, we labeled "*alternative bridging KM practices*" as one of the secondary codes referring to the way DevOps teams support continuous availability and reliability when the main tools and technologies are

down. Inspiring from the data, we introduce the concept of *alternative bridging KM practices* to explain how DevOps teams bridge the gap when they are not able to use predefined and usual KM practices.

In this study, we analyzed data through a Computer Assisted Qualitative Data Analysis (CAQDAS) approach, using Dedoose, which is a web-based application. The CAQDAS approach offers tools that assist with qualitative research (Talanquer, 2014). Dedoose facilitates the process of coding and analyzing by organizing data, searching and retrieving information, interpreting the data, and managing various tasks within a single software. Given that this study follows the inductive approach to build a theory, we found Dedoose beneficial in that it allows us to stay close to the data (Talanquer, 2014).

3.4 Research Validity Criterion

3.4.1 Accuracy of information obtained from interviews

In the grounded theory approach, a comparative analysis is necessary before a conclusion can be drawn (Charmaz, 2006). Here, the comparisons were made across 9 DevOps team members in 3 different organizations. The grounded theory approach requires that the data be verified, and we address it through multiple sources to check for data veracity (Charmaz, 2006; Ridder, Miles, Huberman, & Saldaña, 2014). In this study, we used content which are available on companies' website to verify interview data about the size of the company, configuration of departments, DevOps activities, and technologies used by teams. Information about FinCo., for instance, is available through the several magazine articles published by the company that reflects some aspects of KM in the whole organization, as well as in its DevOps team:

> "You can also read more about this group and its efforts here in this link" (FinCo.P01; FinCo.Document1)

In addition, a few documented articles and books suggested by the managers at FinCo. and DashCo. were also used to corroborate the interviews:

> "Google's has a particular set of best practices and that they've sort of branded as SRE in a book that you can follow." (DashCo.P03; Murphy et al., 2016)

> "I can share a paper with you about DevOps. This is exactly our goal. The comprehensive Infinity model. This model is general and compatible with all DevOps teams in every organization, I think!" (DashCo. P01; ATTLASIAN, 2018)

These documents are publicly available on the company's website and on the public domain. Thus, the information obtained from interviews can be seen as reasonably accurate when we compare the participants statements with the actual KM and DevOps activities in the organization.

3.4.2 Dependability

Given that dependability reflects the consistency in the process of the study and the stability of findings over time (Miles et al., 1994), this study describes the research steps taken from the start of the research project (see Figure 4). Moreover, dependability involves participants' interpretation of the study depicting that all data are derived from the interviews (Korstjens & Moser, 2018; Moser & Korstjens, 2018). As such, this study includes the exact statements of the participants in quotes in the research findings chapter to demonstrate the dependability.

3.4.3 Credibility

Credibility refers to identifying if our research findings are acceptable for participants and the readers (Miles et al., 1994; Yin, 2017). This study will address the credibility by including the rich description of each case. Comparing the initial framework with the results could also strengthen the credibility of this study (Miles et al., 1994). Other than that, triangulation of data was done through using different data sources and multiple sites (Miles et al., 1994).

3.4.4 Confirmability

This criterion refers to the identification of researcher's biases. To support the reliability of this study, we used an inter-coder reliability approach in which two coders (principal investigator and two supervisors) coded parts of interviews and compared and discussed their results (Charmaz, 2006; Ridder et al., 2014). First, the principle investigator coded the interviews to label the primary codes and then attempted to summarize them into secondary codes. Then each author separately coded a section of data that was rich in codes with the goal of preventing biases and finding more emerged codes. After getting agreement on all steps of primary, secondary, and aggregated codes, the possible relationships between our codes within and across cases and as well as extant research were discussed. By inter coding reliability approach and having an agreement

of three authors on codes, this study makes the readers sure that the results are based on the real data rather than on one researcher's perspective.

3.4.5 Transferability

The transferability judgment refers to the degree to which the findings of this study can be generalized and transferred to another research context (Miles et al., 1994). To meet this validation criterion, we provide descriptions of participants (see section 4) as well as the research process (see Figure 4) to give the readers the ability to assess if they can relate the findings from this study to their own settings. Moreover, this study conducted multiple case studies and investigated commonly used KM practices and technologies in several DevOps teams. Through our multiple case study approach, we provide commentary on whether our results may be pertinent in other cases (Korstjens & Moser, 2018; Moser & Korstjens, 2018; Yin, 2017).

3.5 Research Ethics

This study approved by the University of Ottawa's Research Ethics Board to ensure compliance with the Tri-Council Policy Statement on the Ethical Conduct for Research Involving Humans issued by the Government of Canada. Specifically, informed consent and voluntary participation criteria were addressed by making sure the interviewees were aware of the purpose of the study, the detailed possible risks, and the use of the data (Hennink et al., 2011; Korstjens & Moser, 2018). This was done through the Call for Participation form as well as Consent form provided to participants before the interviews. Participants were asked to read and sign the Consent form. To fully meet this criterion, the participant was given the right to withdraw from the study at any point, even after interviews.

In addition to the above criteria, the safety of participants and professional integrity was another ethical goal of this study (Hennink et al., 2011; Korstjens & Moser, 2018). Respecting the anonymity of companies and people involved in the study was agreed upon from the beginning of the interview. In doing so, the data was stored on a secure computer, only accessible to the student researcher and thesis supervisor. The results and findings are reported in anonymized fashion to safeguard the identity of participating organizations and respondents.

4 Findings

4.1 Participating Organizations

There is a strong call for more qualitative approaches, including grounded theory in exploring the role of KM in DevOps (Nielsen et al., 2017). Grounded theory is derived from data inductively through a theoretical sampling and comparative analysis (Charmaz, 2006; Hennink et al., 2011). According to theoretical sampling, this study uses DevOps personnel as sample choice, which is consistent with the area of the study. 9 in-depth interviews with participants from 3 organizations were used in this study. All interviews satisfied the following criteria:

- The interviews addressed broad areas of both KM theme and DevOps theme rather than focusing on few functions such as technologies and skillsets exclusively.
- The interviews included Dev (Development) Personnel, Ops (Operations) Personnel, as well other pertinent Non-DevOps Middle-Management Personnel who might help to better understand inter-team and intra-team knowledge flows.
- The interviews were conducted with personnel of organizations with separate DevOps departments, teams, or DevOps job titles.

Applying these criteria, this section provides a summary of the three cases investigated in this study.

4.1.1 Case1: DashCo

The first organization is a Software as a Service (SaaS) provider that provides data visualization and dashboarding tools to businesses such as digital marketing agencies. This organization henceforth will be known as DashCo for the ease of identification in this study. DashCo is a medium sized company and five participants were recruited from the DevOps/SRE team in DashCo for the purpose of this study. (Table 6).

The first interviewee describes himself as a developer in SRE team (Site Reliability Engineering). As a mid-level experienced member, he provides rich data concerning the ways that Dev and Ops members apply KM approaches and practices to solve the problems together using appropriate tools and technologies. The second participant was a senior release engineer, and he played a key role in testing new features as well as in helping the pipeline release of code. Accordingly, on one hand he has valuable information from the viewpoint of the operation and on

the other hand, he provides comprehensive information about knowledge sharing practices and processes inter and intra-team.

The third participant describes himself as the SRE Manager. His role is to manage people in SRE team and deal with the back-end problems. Dealing with both people and technological aspects of the team, he provided useful insights on different stages of software delivery and related KM practices and challenges. The fourth participant described himself as SRE team member who was mainly responsible for the health of the system. His inputs were valuable in terms of DevOps tools and their changes over time in DashCo. The fifth participant, though new in SRE team, contributed to our data by providing more information from the Dev team viewpoint. His insights helped to get more practical examples on how new employees can learn required skill sets in such an agile team.

4.1.2 Case2: FinCo

This organization is a financial, software, data, and media company. As a large sized company, it provides financial software tools and enterprise applications such as analytics and equity trading platform, data services, and news to financial companies. For the purpose of identification referred to as FinCo. Two participants from this organization were recruited for this study. The first interviewee was a head of Developer Experience (DevX) team. As part of his job, he deals with developers to ensure that the developers can work productively. He provided insightful comments on how his team established consistent and intuitive interfaces, as well as clear documentation, so developers can adopt new workflows and get the most value with the minimum effort in FinCo. Second interviewee was manager of the market data capacity and latency team. He discussed how the efforts of the other participant as manager of SRE team and lead of DevX team trickled down into engineering and operation organization.

4.1.3 Case3: ConCo

This organization provides DevOps entities with solutions to solve their technology problems and improve digital service availability. This small sized benefits this study in that it not only houses its own DevOps teams and job titles but also consults other DevOps organizations by analyzing their DevOps activities and by implementing customized DevOps solutions. Its customized solution becomes a critical part of SRE implementation and drive collaboration, velocity, visibility throughout organizations. For the sake of ease, this organization will be known

as ConCo. Two participants were recruited from this organization. As part of the DevOps team in ConCo, these two interviewees provided valuable data on how Dev and Ops team work on various solutions and projects through collaborative activities.

Moreover, since they actively give insights to other DevOps entities, they provided comprehensive information about tools, technologies, and techniques that are widely used in many DevOps organizations. They helped us understand weaknesses and strengths of current KM technologies in DevOps teams by presenting their solutions during the interview and also in separate sessions. The first one is a solution architect and senior software consultant with broad knowledge on current DevOps approaches. The second one introduced himself as Mid-level DevOps team developer with detailed information on how their team interacts with each other in such an agile environment.

The summary of information of above participants is provided in Table 6.

Table 6: Participant Information Matrix

Experience Level / ID	*DashCo.*	*FinCo.*	*ConCo.*
Manager	DashCo.P01	FinCo.P01	
		FinCo.P02	
Senior Level	DashCo.P02		ConCo.P01
	DashCo.P03		
	DashCo.P04		
Mid-Level	DashCo.P05		ConCo.P02

4.2 DevOps KM

In this study, the enablers and characteristics of KM practices as well as related technologies in DevOps are defined and described through the grounded theory coding procedure. Drawing upon a combination of the review of the DevOps and KM literature as well as an analysis of DevOps teams' interviews, this section attempts to provide a descriptive account of KM practices that play a role in improving DevOps performance.

We found that DevOps teams possess what we call the *perception of KM's significance* which leads them to engage in *DevOps KM practices* characterized by *people-centered KM practices, technology-centered KM practices,* and *alternative bridging KM practices*. These DevOps KM practices highlight the role of KM as enablers of DevOps performance. Appendixes

7.8, 7.9, 7.10, and 7.10 present the case evidence for each element of the final model, and Table 8 provides full information from one exemplary case.

In Figure 5, the three orders of data analysis of this study are structured and portrayed as a data structure. As mentioned in the previous section, "Gioia approach aggregates the overall coding output into a grounded theory model that depicts the dynamic relationships among the emergent concepts." (Chandra & Shang, 2019, p. 12). Definitions and examples of the second-level and aggregate codes are summarized in the code book excerpt provided in Appendix 7.6. Appendix 7.7 summarizes the evidence of multiple case studies around DevOps' perception of KM. Similarly, Appendix 7.8, 7.9, and 7.10 present sample data concerning KM practice and its sub-codes.

In the following sections, we will explain the analysis findings and outline how the primary codes were generated, and then combined into secondary codes and aggregated codes – codes that constitute the elements of our final (emergent) model. We will present the results through three main sections obtained from aggregate dimensions 1) *DevOps teams' perception of KM significance, 2) DevOps KM practices, and 3) DevOps performance*. While doing so, we will also formulate summary propositions that constitute the key take-aways from our findings and provide the basis for the emergent theoretical model of DevOps KM.

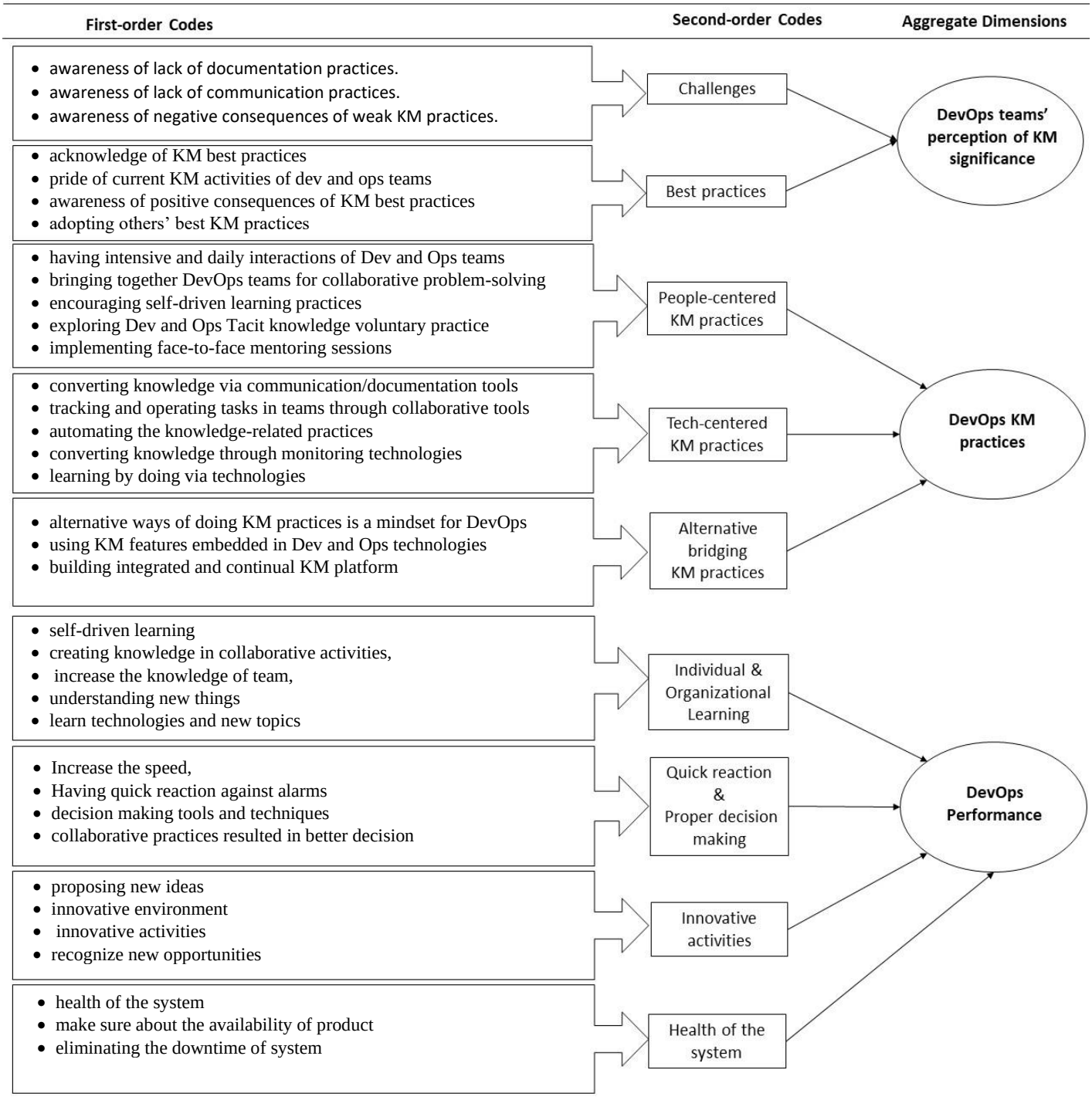


Figure 5: Coding Data Structure

4.2.1 DevOps teams' Perception of KM's Significance

In Figure 5, the first aggregate code we obtained from data is *DevOps teams' perception of KM significance*. The idea that DevOps personnel realize the significance of knowledge management practices is observed in the quotes of most of the participants in this study. This awareness of the importance of KM is directly related to the actions and initiatives of DevOps teams in the KM area (see Orlikowski, 2002 for possession and knowing in practice). Appendix 7.7 presents the case evidence for DevOps teams' perception of the importance of KM practices. The results arise when DevOps teams were asked to explain their view on challenges in knowledge sharing as well as best practices that were being used in their organizations.

In the following, we will explain how participants perceive the importance of KM practices in DevOps which then can be an inspiration for participating in KM practices.

DevOps Perception of KM's significance due to existing challenges

The participants mentioned the extent to which they realize the importance of KM by speaking about the weaknesses (challenges) of KM practices and knowledge sharing activities. Participants frequently mentioned how they suffer from a lack of knowledge-related practices in their team. In *DashCo.P02's* words:

> "lately we had layoff, and somebody left the team and their knowledge left the team. You do not know what you did not know until personnel leave. So, having known we've gone through this were made my effort to increase the coverage of documentation of what we had in Confluence".

According to *FinCo.P01*:

> "we didn't have enough knowledge sharing and we didn't have enough training provided so that teams are struggling with writing Jenkins pipelines…luckily with the new KM platform going out after that, change has been kind of resolved."

Following Dedoose analysis reports that most of the managers signify the negative consequences of weak documentation practices in their team. As shown in the Figure 6, while most participants acknowledged the negative consequences associated with lack of documentation practices, the

managers who were responsible for building a strong team with a shared vision were especially vocal about this issue.

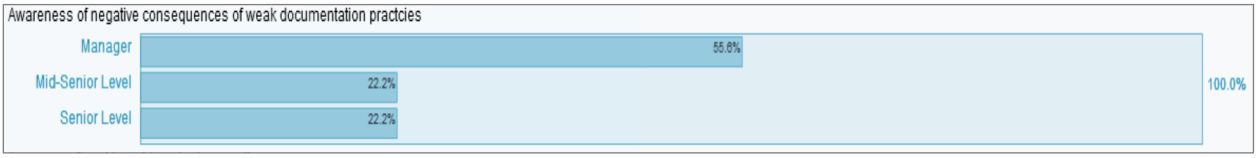


Figure 6: Dedoose Analysis: code as awareness of negative consequences of weak documentation practices and descriptor as experience level

Such repetitive concerns shaped our first-order codes as: "*awareness of lack of documentation practices*", "*awareness of lack of communication practices*", "*awareness of negative consequences of weak knowledge-related practices*". To unify these primary codes, we interpret them as a second-order code *"awareness of challenges"* (see section 3.3 for detailed data analysis process).

Here challenges are viewed as ways to understand the importance of KM practices. We found that DevOps personnel indicate the need for the knowledge management practices when they talk about their challenges in the absence of KM practices. Appendix 7.7 is a sample data from interviews pertaining to DevOps teams' perception of KM significance. This Appendix contains various examples of cases in which DevOps managers and engineers claim that they have already experienced the negative consequences of weak KM practices and realized the importance of KM practices so, put more effort to adopt structured KM practices (e.g., searchable documentation, informal meetings, communication tools, etc.).

DevOps perception of KM's significance due to best practices and standards

The second interview question that leads us to understand the importance of KM practices from the viewpoints of DevOps teams is whether they have best practice frameworks or standards as well as teams' plan for running KM practices in organization and improving the current status of KM in their DevOps team? This question is addressed specifically by leaders and managers who ranked their teams' performance in terms of KM activities by comparing themselves with those companies who successfully established the KM platform for their DevOps/SRE teams (e.g., Google). They often expressed how successful stories shape how they think about the necessity of implementing KM practices:

> "Google's has a particular set of best practices and that they've sort of branded as SRE in a book that you can follow. So, they have sort of a prescribed way of how the cycle is made and Google was one of the first companies to realize that you need people with these sort of cross-pollinated skill sets who can share their experiences." (DasCo.P03)

> "We are increasingly presenting at conferences too. We are aware of different solutions and want to adopt best practices from others, in addition to contributing our best practices to the industry." (FinCo.P01)

DevOps teams explained in detail and with pride how their unique KM best practices helped the firm overcome challenging times. For instance, DevOps personnel acknowledge that they realize the importance of KM practices because they were not able to survive past perilous times without well-organized documentation or communication practices (e.g., regular meetings and reports sessions). We labeled these raw data as first-order codes: *acknowledge of KM best practices, pride of current KM activities of dev and ops teams, awareness of positive consequences of knowledge-related best practices, adopting others' best KM practices.* The first-order codes summarize the key evidence of quotes and show how the data give meaning to KM practices by placing the awareness of today's risk as well as challenges from the past in perspective. Then, as Figure 5 shows, we collapsed these first-order codes into second-order code as *awareness of best practices* (see section 3.3 for detailed data analysis process). Here, awareness of best practices is viewed as a way to understand the importance of KM practices. Appendix 7.7 is a sample of data from interviews pertaining to DevOps teams' perception of KM significance. This Appendix shows how the awareness of KM best practices and their positive consequences motivate DevOps to adopt wide ranges of KM practices from inside and outside the company within their teams.

According to the above explanations, both the *awareness of best practices* and the *awareness of challenges* can be abstracted into "*DevOps' perception of KM significance"* as the aggregate dimension. This study thus introduces the concept of *DevOps' perception of KM significance* as a new theoretical construct that helps explain how DevOps members can be motivated to pursue relevant KM practices (we will discuss types of KM practices in the next section). As shown in Figure 7, perceptions of KM significance were discussed by participants across all companies.

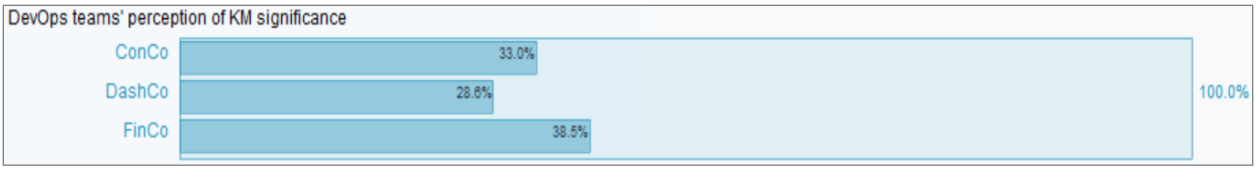


Figure 7: DevOps teams' perception of KM significance in companies

To address the relations between *DevOps teams' perception of KM significance* and *DevOps KM practices*, this study provides following Dedoose analysis to show that these two codes occurred in the same excerpt.

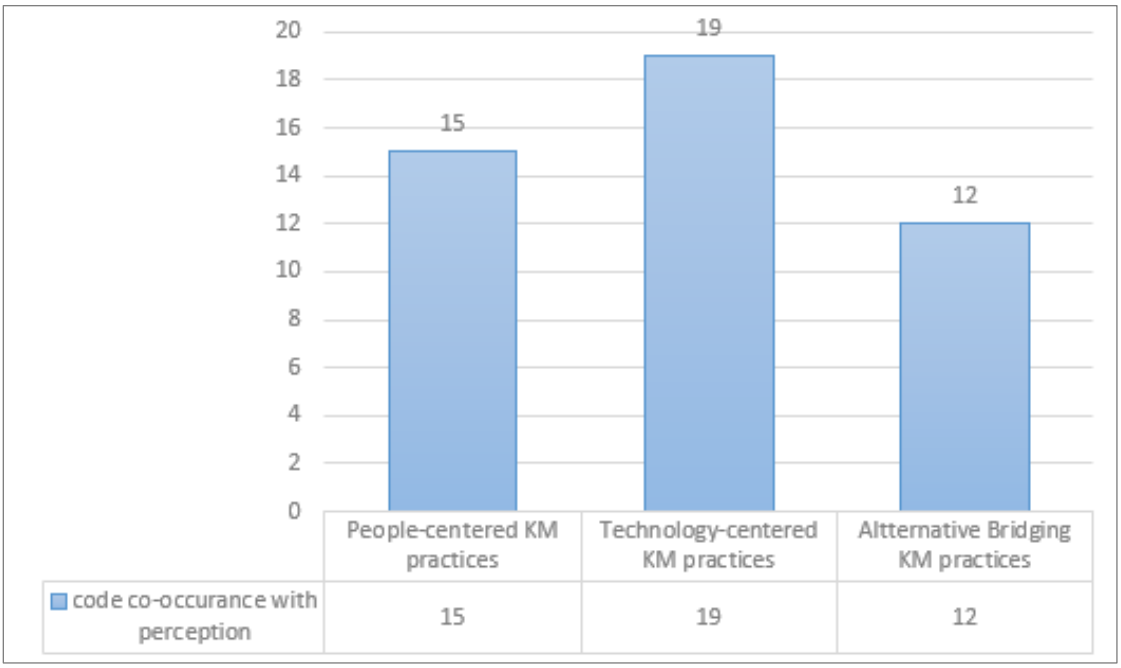


Figure 8: Code Co-occurrence: DevOps teams' perception of KM significance & DevOps KM practices

As depicted in Figure 8, DevOps personnel mentioned how they increasingly engage in KM practices once they realized the importance of KM practices indicating that the perception of KM significance leads DevOps teams to participate in KM practices. Table 7 presents the case evidence for each of these two codes to show the relationship between them.

Table 7: Exemplary evidence of relation between DevOps teams' perception of KM significance with KM practices.

Summary of key evidence - Perception of KM Significance	Summary of key evidence - KM Practices	Quotes from participants
Awareness of best practices: it is very effective because it's part of a workflow.	**Alternative bridging KM practices:** documentation and the communication within DevOps tool... we're trying to do as much as possible.	"documentation and the communication within DevOps tool, it is very effective because it's part of a workflow, we're trying to do as much as possible" (FinCo.P01)
Pride of current KM activities of dev and ops teams: This approach is new and very useful for me. I could say such mentoring is one of the strengthen of current KM practices.	**People-Centered KM practices:** employees can ask questions from each other. They arrange a short meeting to talk about that topic.	"Based on this database, the employees can ask questions from each other. They arrange a short meeting to talk about that topic. This approach is new and very useful for me. I could say such mentoring is one of the strengthen of current KM practices." (DashCo.P01)
Awareness of negative consequences of weak KM practices: somebody left the team and their knowledge left the team.	**Technology-Centered KM practices:** …my effort to increase the coverage of documentation of what we had in Confluence.	"lately we had layoff, and somebody left the team and their knowledge left the team. You do not know what you did not know until personnel leave. So, having known we've gone through this were made my effort to increase the coverage of documentation of what we had in Confluence" (DashCo.P02)

The above examples expressed by personnel of DevOps, lead us to the proposition that DevOps teams realize the value of KM practices that move them toward practices by which knowledge can be managed. Thus:

P1:

DevOps teams that recognized KM significance frequently engaged in KM practices. The perception of KM importance helps motivate DevOps' future KM practices.

4.2.2 DevOps KM Practices

Another group of first-order codes was related to DevOps KM practices that lead us to develop a theme around this concept. The data obtained from the interviews show that by understanding the importance of KM practices, DevOps teams attempt to participate in the web of

practices that we called *DevOps KM practices.* DevOps teams believed that they use idiosyncratic KM practices that are characterized by combinations of *people-centered KM practices*, *technology-centered KM practices*, and *alternative bridging KM practices* (Figure 9).

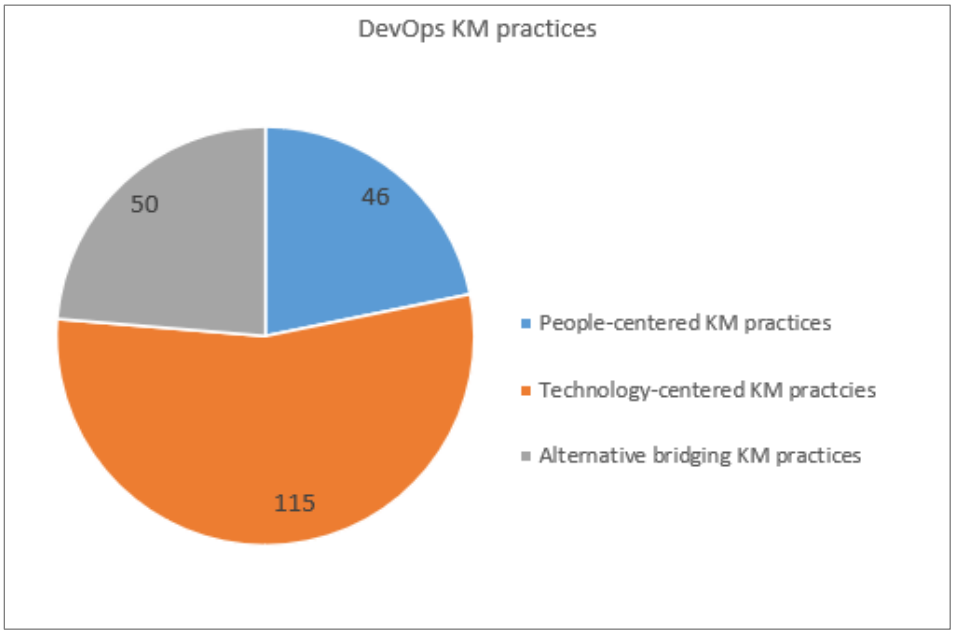


Figure 9: Characteristics of DevOps KM Practices

According to the statements of interviewees, these KM practices then could drive the DevOps performances. This section will highlight some characteristics of each of these KM practices that are captured in the quotes of DevOps teams.

4.2.2.1 *People-centered KM Practices*

The review of data shows that there is an obvious evidence of people-centered KM practices in DevOps team with the purpose of converting an individual's tacit knowledge into a tacit knowledge of other individuals as well as explicit knowledge. Consistent with the practice perspective of KM discussed in section 2.1.6, we found that in conversion of tacit to tacit and tacit to explicit knowledge, DevOps teams' members (people) are the key elements who possess the knowledge and perform the practices. According to interviewees' statements, each of DevOps team member with different expertise level has unique knowledge and thus can carry out the knowledge practices. This view helps two groups of Dev and Ops working as a single team to perform tasks. The data from three cases highlight the unit that possesses knowledge by using

terms such as “team”, “people”, “you”, “I”, “we” show to what extent DevOps consider their personnel as essential elements and initiatives of KM practices. To mention some:

> “So you ask a question and chat, you find out things, you post a simulated question on Stack Exchange…and you can go into all of our problems solved this way and you just follow what's there” (FinCo.P02)

> “I have to do some DevOps operations; I have to push code and make up to write code first and then push it to a repository.” (ConCo.P01)

> “platform will generate file for me, but I am responsible to populate the content.” (FinCo.P01)

This pattern is consistently expressed by all participants which shows that DevOps personnel introduced themselves as essential elements for tracking tasks. Thus, we labeled these raw data as first-order code: *Dev and Ops teams are key responsible agents in operating tasks.*

After developing an understanding of the significant role of DevOps teams in KM practices, our data further reveals the way in which knowledge practices get done by DevOps teams. To address this aspect of practice perspective, DevOps teams were asked to explain the processes and practices that are needed for the fast-paced nature of DevOps. DevOps teams argued that they were deeply involved in some critical KM practices which shape our first-order codes: *having intensive interactions of Dev and Ops teams, bringing together Dev and Ops teams for collaborative problem-solving*. This way managers of all three cases encourage their DevOps teams to convert their tacit knowledge into explicit knowledge. There is a consistent pattern in all of these DevOps teams that they increasingly participate in different types of cross- teams meetings and standups to be able to make sure that knowledge is shared between whole team plus any customer that might be downstream consumers and upstream providers. DevOps teams express a very strong belief in the importance of such face-to-face gatherings between Dev and Ops teams. This pattern also indicates that these meetings focus on bringing together Dev and Ops teams to have collaborative troubleshooting.

All three cases share many similar KM practices when they talked about different ways of problem-solving. Although different in name, all such KM practices focus on bringing together Dev and Ops teams to support collaborative troubleshooting. For instance, DashCo. Organizes what are called Sync Up meetings once a week in which PM/UX/Dev teams (product management/

user experience/development) discuss technical issues specifically in design phase. Moreover, another common meeting is After Action meeting which is among DevOps personnel to investigate the problems as well as new insights for future use. In Lunch and Learn meetings, individuals in DevOps prepare a video about what they did and present it to the entire team.

Following the way DevOps teams perform KM practices, we found interesting methods they pursuit to target the interests of personnel that allow individuals to be more likely to convert their tacit knowledge into explicit knowledge. Though the major focus is to spread technical knowledge through compulsory meetings (e.g., After Action, Sync Up, etc.) it is aided significantly through informal and voluntary meetings. A sample of quotes from DashCo. can be found here:

> "There are some professional groups called Interest Group such as data science group and testing group. They meet each other every week or two weeks to discuss the technical issues. The interesting thing for most of these opportunities is that they are not compulsory, rather, it is up to the members to participate in these activities." (DashCo.P01)

> "A lot of training is also done is self-driven. Somebody will investigate a new technology and then through meetings or documentation convey that information to the team." (DashCo.P02)

At FinCo, DevOps teams operate on a similar belief that Guild and Chant program is a cross-teams approach for a group of people who have a common interest on certain technology or topics to have regular meetings and meetups and share their ideas.

> "This is another way to facilitate cross-team collaboration, which is not necessarily easily achievable by a traditional organizational knowledge sharing structure." (FinCo.P01)

These voluntary KM practices lead us to capture first-order codes as: *exploring Dev and Ops tacit knowledge through voluntary practices* and encouraging *self-driven learning practices.*

In addition to the above, another KM practice is mentoring which is highly recommended by DevOps personnel in this study from management level to development and operation engineers. At DashCo. and ConCo., they have dedicated on-site senior people who are responsible for real-time training and problem-solving. As with most interviewees in this study, we found that this face-to-face mentoring is crucial to DevOps' success in that they need something to react

quickly and a face-to-face communication works well in such a changing environment. This led us to develop first-order code as *implementing face-to-face mentoring sessions*. The highlighted quotes are only meant to be a representative sample; Appendix 7.8 provides a broader sample of quotes that identified potentially relevant KM practices.

The final set of first-order codes (i.e., *having intensive and daily interactions of Dev and Ops teams, bringing together Dev and Ops teams for collaborative problem-solving*, encouraging *self-driven learning practices, exploring Dev and Ops Tacit knowledge voluntary practice, implementing face-to-face mentoring sessions)* reflects all of the people-centered KM practices as given by DevOps personnel in this study. In nutshell, the above results first show that individuals in Dev and Ops teams are key components of KM practices who possess the knowledge. Referring to the practice perspective of KM in this study, the results obtained from the data also show how we can integrate the teams that possess the knowledge and the way they perform the knowledge-related practices. The above knowledge practices facilitate the conversion of tacit knowledge into tacit and explicit knowledge and led us to group them as *people-centered KM practices (*as second-order code) that comprise DevOps KM practices in our final model. We found that this pattern (code) occurs in all three companies (Figure 10).

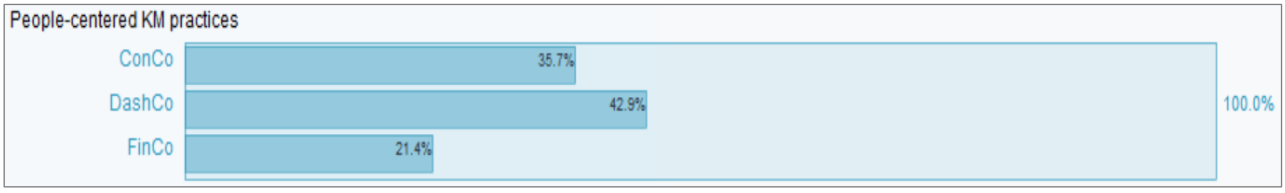


Figure 10: People-centered KM practices in companies

Also, as shown in Figure 11, there were multiple instances of participants discussing people-centered KM practices while talking about the significance of KM.

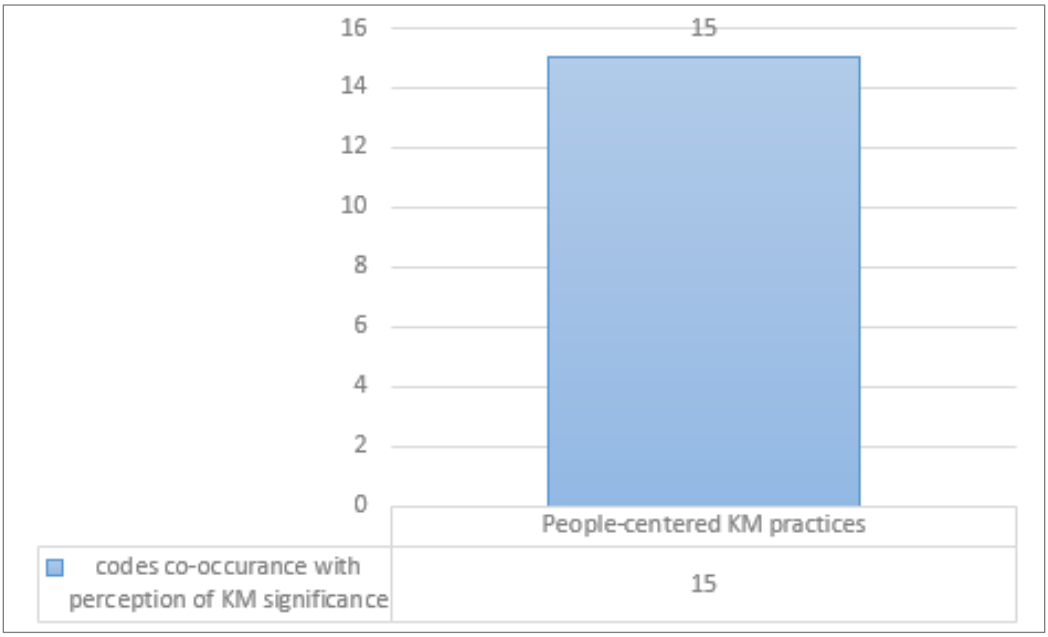


Figure 11: Codes co-occurrence: DevOps teams' perception of KM significance & people-centered KM practices

To summarize:

P2:

DevOps teams that recognized KM significance frequently engaged in people-centered KM practices.

4.2.2.2 *Technology-centered KM Practices*

The second type of practices that comprised DevOps KM practices concerned technology-based practices (technological infrastructure for KM as codification). Figure 9 shows that most DevOps are geared toward these type of KM practices. Again, here the data highlighted the role of DevOps teams as a unit that possesses the knowledge and initiates the practices, however, this section reports how technologies enable DevOps to convert individual's tacit knowledge into explicit knowledge, explicit knowledge into new sets of explicit knowledge, and explicit knowledge into tacit knowledge of individuals. These types of knowledge conversions were identified earlier from the KM literature (see Nonaka & von Krogh (2009)). Next paragraphs focus

on first-order codes obtained from data around technology-based KM practices. Please note that all the recommended DevOps tools and technologies are summarized in Appendix 7.5. Additionally, Appendix 7.9 presents more exemplary evidence of quotes of technology-centered KM practices in DevOps.

We noted different kinds of DevOps technologies by which knowledge can be managed. Interviews with DevOps personnel highlighted the importance of individuals' tacit knowledge and the way individuals use technologies to convey their knowledge to the entire team. Referring to interviewees statements what makes DevOps different is that the fast and changing environment of DevOps leads them to learn new things themselves and come up with a solution for real-time problems. Therefore, regular meetings are not enough for such real-time troubleshooting. Technology-based KM practices thus are needed to speed up the interconnections. We found that the likelihood of knowledge transfer between individuals in DevOps is increased when a person uses technology.

We found that performing KM practices with the purpose of converting tacit knowledge into other types of knowledge is important in DevOps, giving rise to the notion of using KM technologies. In one of our cases, for example, the Instant Message System which is an in-house chat room for inter-team communications plays a significant role in FAQs. Similarly, Slack provides different private and public channels that enable individuals to convey their own knowledge to their colleagues without the need to meet them in person.

> "Slack is where people live and breathe these days. It's the fastest way to communicate" (DashCo.P03)

Microsoft Teams and Stack Exchange are other examples of communication platforms expressed by interviewees when they were asked to explain how they communicate inter-team and intra-team.

A similar pattern is observable for the storage of tacit knowledge of DevOps teams' members via various intranet tools. There is a general desire for more documentation practices during all software development stages for future use. At FinCo, for instance, structured KM platform allowed DevOps teams to document the software build and installation instructions, maintain operator troubleshooting guide, prepare training materials to aid users and testers,

training material for installation and configuration, and to update the materials with each major software release.

None of the two other cases, reported such a comprehensive in-house documentation platform that can support the integration of documentation in all stages of software development from design to production phases. As DashCo.P05 formulates:

> "we do document that describes the design history, but not a troubleshooting guide kind of documentation because documentation is really hard to keep up to date, everything is changing all the time, you can spend a lot of time updating your docs for no value because you still have to figure out whatever the next problem".

Interviewees at both cases did indicate a need for more guidelines for use of shared documents between Dev and Ops teams.

Documentation of individual and collective knowledge mainly takes place via open-source technologies. Confluence, a popular documentation tool, was frequently mentioned by DevOps teams indicating how individuals store their own tacit knowledge for future technical reviews. Talking about the documentation practices he implemented in DashCo, participant DashCo.P02 introduced Confluence as a primary library of how to do something:

> "Confluence, I would say is the primary library of how to do something, there is document in there and also let the documents for plans for what we want to do in the future. It also has checklist, in a role if you have something to do to release the product for example you have to go through the checklist to make sure that those points have been satisfied before you release the product to the customer."

Moreover, Architecture Decision making Records (ADR) was another way of documenting the troubleshooting processes in which DevOps engineers fill out a form and describes how they solve a problem:

> "we have an ADR process which is called Architecture Decision making Records. And you basically fill out a document about what the problem you're trying to solve, what the possible options were and why you chose the option…it's good to have some idea of why we made a decision, because, you know, five years from now when it's a whole new team,

> different people work here and they're trying to figure out why we do this. That document is very helpful." (DashCo.P04)

Similarly, Dev teams state that engineers in Dev team who develop the code in code management tools (e.g., GitHub) are responsible to document and update their changes.

> "we create code, it gives you a box says to describe the changes, describe what you did within the GitHub environment." (ConCo.P02).

This enables test engineers later to review any impact and minimize the risk to the product. Additionally, Quality Assurance teams are responsible for documents test results and maintain test data files (TDF) for future assessments. Documents created during software development phases also stored and shared via SharePoint intranet portals. (see Appendix 7.8 for more exemplary evidence in this area). These practices set up our first-order code in terms of technology-based KM practices as *converting knowledge through communication/documentation technologies.*

In all of the responses the ultimate objective of using KM tools is so clear. DevOps teams try to keep the same level of understanding of problems among development and operations teams. Based on the data obtained from cases, leaders spread the individual and collective knowledge using collaborative technologies while working on a certain project. The data from the interviews suggest that issue tracking tools such as YouTrack benefits DevOps teams by creating custom workflow and enabling them to tag and track responsible persons. At DashCo. This tool is widely used by all DevOps teams to support the input of multiple people who are working in one single project and solve the bugs collaboratively:

> "U-track and Jira, they provide an agile environment for tracking the problems. The interesting thing is that we can track the responsible person through the You Track for each assigned task. It shows the processes, the estimated time. We have to document the processes through this software even small tasks." (DashCo.P01)

Project management tools such as Jira are similarly used to exposed DevOps to the latest practices and updates:

> "we have TFS (Team Foundation Server), we use it for project management, also Jira. we have to go back sprints…and as long as you're getting your tag on updating your tasks regularly, so people know where you are." (ConCo.P01)

These technologies allow the developers and operations to combine or edit their knowledge in large-scale databases (i.e. knowledge combination). We summarized these KM practices as *Tracking and operating tasks in teams through collaborative technologies*, is shown in data structure, Figure 5, as first-order code.

Another first-order code reflects DevOps concerns when they talked about the team plan in using different technologies and tools. The companies exhibit great commonalities in the automation of tasks. Having this mindset, DevOps use technologies to minimize manual practices in terms of KM. For instance, Ansible which is an open-source software provisioning, configuration management, and application-deployment tool enables DevOps teams to automate inventory management, centralize automation execution, and control job scheduling and role-based access. In doing so, it facilitates team interactions by eliminating repetitive tasks. Equally important, all companies in this study make good use of automatic alerts and testing bots. DevOps personnel who are responsible for health of the system are notified 24/7 via SMS or automate Slack alerts about back-end system alarms:

> "we have to be able to work remotely because we support the product like 24/7. So, if anything happens at night, we are all up we need to know how to communicate things very well". (DashCo.P04)

We label these practices as *Automating knowledge-related practices.*

In addition to the above, data shows that one of the critical technologies for DevOps teams is monitoring tools supporting long term and short-term business trends:

> "we need more specific monitoring tools rather than the other departments. We need to gather real-time data and conduct the troubleshooting processes online." (DashCo.P01)

Accordingly, data suggest that monitoring tools are important in conversion of explicit knowledge into new sets of explicit knowledge. For instance, the distributed data (explicit knowledge) can be collected automatically and then can be visualized for further use (explicit knowledge).

> "I have added many monitoring tools. For instance, in the past, we had many problems and we did not know how to prioritize and solve them in our limited time. I made a dashboard that can make a query from our database, it gathers all the bugs that people reported and it finds the factors that cause the most problems in our system so we can monitor those highlighted bugs in charts or stuff like that." (DashCo.P01)

This pattern mentioned by DevOps personnel when they talked about collaborative troubleshooting. For instance, there are many tickets and feedbacks submitted by Ops teams and need to be solved by Dev teams. This feedback gathered from different sources, processed and integrated via dashboards into new sets of explicit knowledge.

> "when bunch of teams are working on a single outage, we can create links to search and pass it around on Graphene (monitoring tool)" (FinCo.P02)

As such, monitoring tools allows Dev teams to prioritize these tickets, create alerts, monitor the health of the system, and observe different trends (new sets of explicit knowledge). These practices were collapsed as the first order code of *converting knowledge through monitoring technologies.*

Last but not least, data shows that technologies allow DevOps teams to engage in KM practices and convert the explicit knowledge from different sources into tacit knowledge of individuals. This can be done most naturally by learning via online sources:

> "developers' best friend is Google because we search for everything. Somebody did something out there that is always similar to yours. we are trying to discover brand new things" (ConCo.P01).

At FinCo. there is an online internal university with various online courses, tech talks, and workshops by which individuals can improve their skills and enhance their knowledge. Examples also include the cross-functional collaborative KM practices when developers build a code in Pull Request (PR) GitHub environment and tag other people who are knowledgeable in that area. This way they will review the new code together and assess the possible risks, so there is a common belief that people learn a lot in PR. According to the analysis, development and operation communities could execute and internalize explicit knowledge by following tasks: documenting software build, preparing and maintaining trouble-shooting guide, working with integrated product teams, testing, and detailed tracking. These practices allow individuals in DevOps teams to

assimilate knowledge in order to convert the explicit knowledge into tacit knowledge (i.e., knowledge internalization). Thus, the first order code developed here as "*Learning by doing via technologies"*.

In sum, the first order codes that integrate the KM practices and relevant technologies are: *converting knowledge through communication/documentation technologies, tracking and operating tasks in teams through collaborative technologies, automating the knowledge-related practices, converting knowledge through monitoring technologies,* and *learning by doing via technologies.* As Figure 5 shows, these practices then collapsed into one single theme as *technology-based KM practices* which comprise DevOps KM practices. (see section 3.3 for detailed data analysis process).

The code-co-occurrence of above technology-based KM practices and perception of KM significance is presented in Figure 12. It depicts that DevOps teams recognized the importance of KM that leads them to engage in technology-centered KM practices.

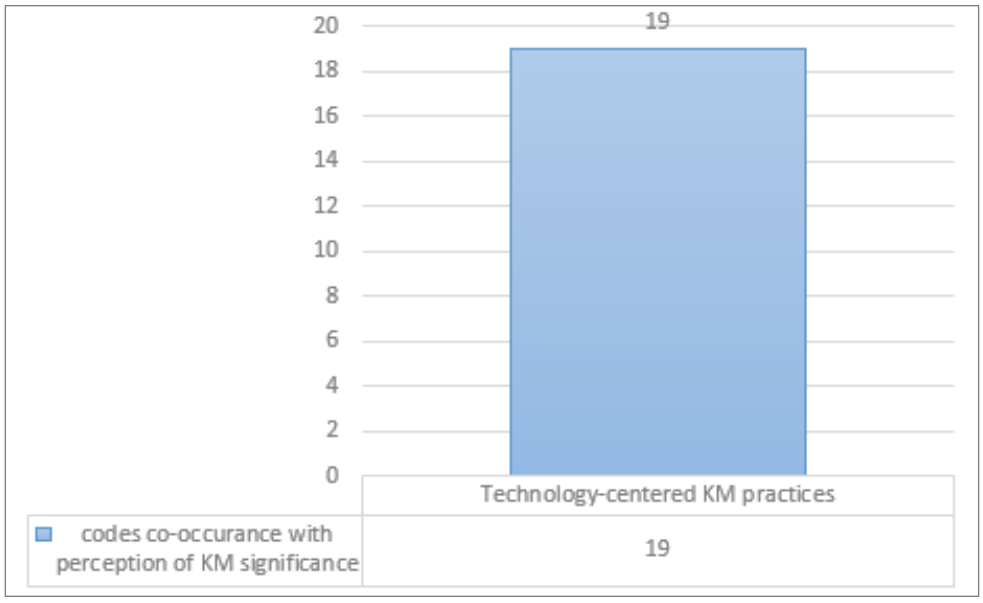


Figure 12: Codes co-occurrence: DevOps teams' perception of KM significance & Technology-centered KM practices

Relying on the above evidence, we present the third proposition:

P3:

DevOps teams that recognized KM significance frequently engaged in technology-centered KM practices.

Deep involvement of DevOps in using technology is not only important for DevOps to develop an understanding of the teams' latest status, but it also facilitates their collaborative tasks that will motivate and give meaning to their own future KM practices.

4.2.2.3 *Alternative Bridging KM Practices*

Beside people-centered and technology-centered KM practices, data presents the view that one of the most important practices in managing the knowledge is the way DevOps teams support continuous availability and reliability when the aforesaid tools are down or when there are some problems with people-centered and technology-centered KM practices. Inspired from the data, we introduce the concept of *alternative bridging KM practices* to explain how DevOps teams bridge the gap when they are not able to use predefined and regular KM practices. This section explains how first-order codes emerged from the interviewees' statements and how data lead us to develop theme around this new concept. Appendix 7.10 provides exemplary evidence of quotes concerning this area.

Our data analysis reveals that the idea of having alternative ways of regular practices is a mindset for DevOps teams and can be seen in various daily activities. These practices allow development and operation teams to guarantee that the software delivery system does not face any problems due to a lack of communications or documentation practices. The view of having alternative tools and techniques, though different in each case, is similar in all cases in that DevOps teams always introduce themselves as prepared enough for such challenging situations:

> "what if slack is down? we're using Google Hangouts, we also made sure that all of our self-healing bots, which usually we control through slack can be done and we can either talk to them via rest API or we can send command line notes to them. We use Confluence; it's just as a way to bridge the gap when we aren't able to fully automate things." (DashCo.P03)

Appendix 7.10 provides quotes about the importance of alternative KM practices. Thus, these data led us to set our first order code as *alternative ways of performing KM practices is a mindset for DevOps*.

Our result suggests that very important alternative KM practices in terms of creating, sharing, storing, and utilizing knowledge are hidden KM features (e.g., chat box and repository) that are embedded in technologies used for code development, testing, release management, and configuration management. Although separate communication and documentation KM technologies (e.g., Slack and Confluence) are popular between Dev and Ops, KM features within DevOps technologies (e.g., GitHub) frequently highlighted by interviews saying that:

> "documentation and the communication within DevOps tool are very effective because it's part of a workflow we're trying to do as much as possible" (FinCo.P01).

Based on the data, usual communication technologies such as Slack are not enough, as ConCo.P02 engineer puts it:

> "I don't recommend Slack because this only saves like 50 messages".

This weak point can be supported by repository features within DevOps technologies because all conversations between developers, for example, could be stored in repository of GitHub PR (DevOps software development platform) for a long period of time. Moreover, DevOps prefer to use KM features within Git Hub because:

> "It would be hard to email or share such file on network. We put our files on GitHub repository" (DashCo.P01).

Therefore, DevOps teams in this study consider KM features embedded within DevOps technologies as alternative ways to mitigate the weaknesses of usual KM tools. The increasing number of DevOps engineers who suggest KM features shape how we think about first-order code as *using KM features embedded in Dev and Ops technologies*.

Our concept of alternative and bridging KM practices also introduces the way by which DevOps synchronize and integrate all above mentioned people-centered and technology-centered KM practices in a single environment. The data shows how some but not all DevOps can excel themselves by providing an in-house KM platform. At case FinCo., the success of their entire

business model depends on the sophisticated knowledge management, enabled by the in-house KM platform. All personnel are connected to the knowledge management platform for different purposes. Although DevOps teams use open-source communication and collaboration technologies such as Stack Exchange, the shared and documented information of all these technologies are recorded within an in-house KM platform simultaneously for future use. This enables them to convert the explicit knowledge of such open source technologies into explicit knowledge of in-house KM platforms that can be used for business plans and trends. This platform provides access to a wide range of knowledge resources to customers and suppliers. As discussed in section 4.1.2, there seems to be evidence that FinCo. as a large sized company needs to have a sophisticated KM platform to be able to connect various departments:

> "because we are large scale engineering company and we have the need of be able to stitch together different tools…without a platform that glue things together, it is very difficult for different teams to figure out what is the best practices" (FinCo.P01)

We found such a comprehensive KM platform merely in FinCo., however, data indicate that other DevOps teams also put effort to bridge the gap and build a link between different open source KM technologies and techniques in order to integrate workflows and support continuous and reliable KM practices. Appendix 7.10 provides a sample of quotes from DevOps teams that illustrate their viewpoint about the integration of KM practices. The simple example is when DevOps teams explain the way Slack get a query from YouTrack and other software. In DashCo.P01's words:

> "Slack has good integration with the application. The reports, channels, even the certain internal files of visualization can be shared through the slack".

According to ConCo.P01:

> "Microsoft Teams really makes everything into one little thing, it is wonderful everything is integrated.".

The consensus between DevOps teams about integration of KM practices leads us to capture first-order code as *building integrated and continuous KM platform.*

In a nutshell, the above first-order codes are summarized into *alternative bridging KM practices* that eventually comprise KM practices in final model (see section 3.3 for detailed data analysis process). As Figure 13 shows, while participants from all companies talked about such practices, this phenomenon was reported relatively more by DashCo. and FinCo. participants. One possible reason could be that according to data provided in section 4.1, these two companies have more mature DevOps entities and have more developed DevOps practices as compared to ConCo.

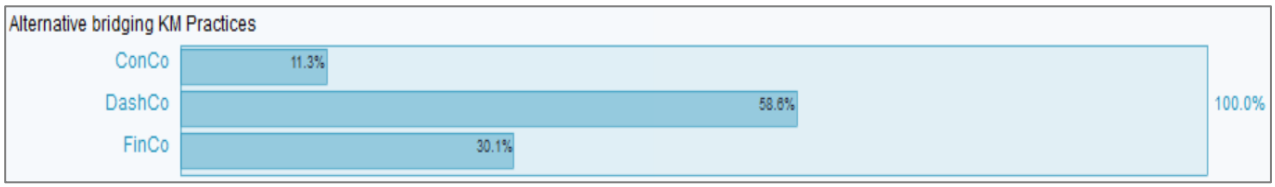


Figure 13: Alternative bridging KM practices in companies

The code co-occurrence of these types of DevOps KM practices and the perception of KM importance is presented in Figure 14.

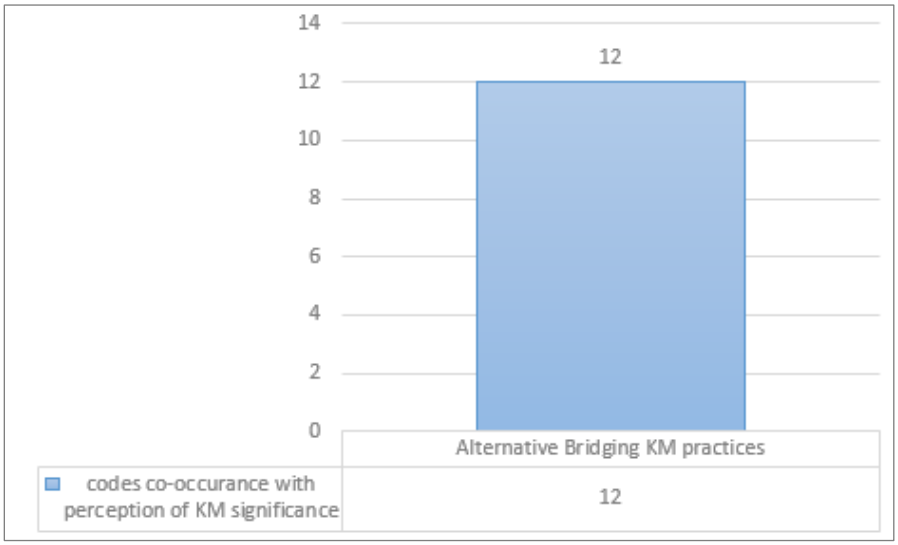


Figure 14: Codes Co-occurrence: DevOps teams' perception of KM significance & Alternative bridging KM practices

Thus:

P4:

DevOps teams that recognized KM significance frequently engaged in alternative bridging KM practices.

4.2.3 DevOps Performance

After developing an understanding of DevOps KM practices that are characterized by people-centered, technology-centered and alternative bridging KM practices this section proceeds to discuss our findings about DevOps performance. According to data, DevOps teams highlights four types of performance outcomes as Figure 15 presents. Furthermore, this section explains how these DevOps performance outcomes may be linked to previously discussed DevOps KM practices.

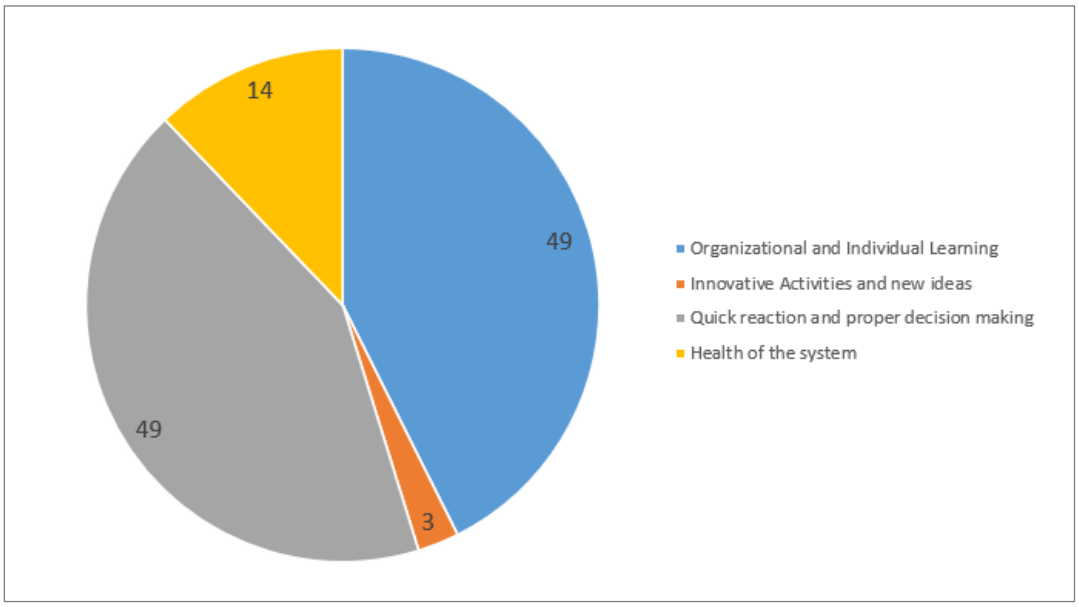


Figure 15: DevOps Performance Outcomes

We drew upon keywords for different aspects of performance identified by DevOps literature (e.g., generative organizational learning (OL)) and then we searched for their possible synonyms (e.g., teach, memorize, find out) and highlighted paragraphs in which DevOps personnel talked about these topics in their own words (e.g., self-driven learning, understand new things). In doing so we capture first-order codes around three main DevOps performance themes: *individual/organizational learning, generating new ideas, quick reaction and proper decision*

making, and *health of the system*. We developed these themes when interviewees talked about the impact of KM practices in their daily work.

Our first-order codes outline DevOps statements that contain keywords such as self-driven learning, create knowledge, increase team knowledge, understand new things, etc. We called them *Individual and organizational learning* as second-order codes that constitute DevOps performance. Individual and organizational learning occurs as a result of involving DevOps members in KM practices and equipping them with technologies they require to convert their tacit knowledge into explicit knowledge and vice versa.

> "Our in-house University, that people internally can go sign up and search and be like, I'm having problems with our internal build system. And here's a workshop that happens once a month. They can go join and figure out how to use and take problems to this workshop to learn and figure out." (FinCo.P02)

> "A lot of training is also done in self-driven, like learning a new technology and then somebody will investigate a new technology and then through meetings or documentation convey that information to the team." (DashCo.P02)

To better address the links between KM practices and related DevOps performance we use Dedoose code co-occurrence analysis. We learn that DevOps frequently highlight the individual and organizational learning as a result of KM practices (Figure 16).

In addition to the above, our data structure shows first-order codes around interviewees' keywords such as speed, quick reaction, decision making tools, decision-making techniques. We labeled these codes as *quick reaction and proper decision-making* (second-order codes) that constitute DevOps performance. DevOps teams direct our attention to these keywords when they explain for example how KM practices such as usual after-action meetings enhance their own efficiency to speed up their decision and to support the health of the system

> "We have also ADR basically, we write a document that explain why we choose this technology or we made this decision over others. We come up with a few solutions we try to compare them, document them and then we have a meeting we discuss that we receive make sense and we all agree." (DashCo.P04)

> "I would have to spend time doing Web applications and then publishing it and then doing other items. So it makes time, you know, it's less efficient that way. So when you separate out the values and tasks between teams it gets quicker." (ConCo.P01)

Another set of first-order codes reflects DevOps statements concerning keywords such as new ideas, innovation, new topics, new. We captured these keywords and categorized them as *innovative activities* (second-order codes) that constitute DevOps performance. Data suggest that KM practices (e.g., meetups and collaborative troubleshooting) lead DevOps to innovative activities wherein DevOps engage in multiple acts of proposing new ideas. This pattern is observable specifically when individuals take part in KM practices that allow them to convert their tacit knowledge into explicit knowledge.

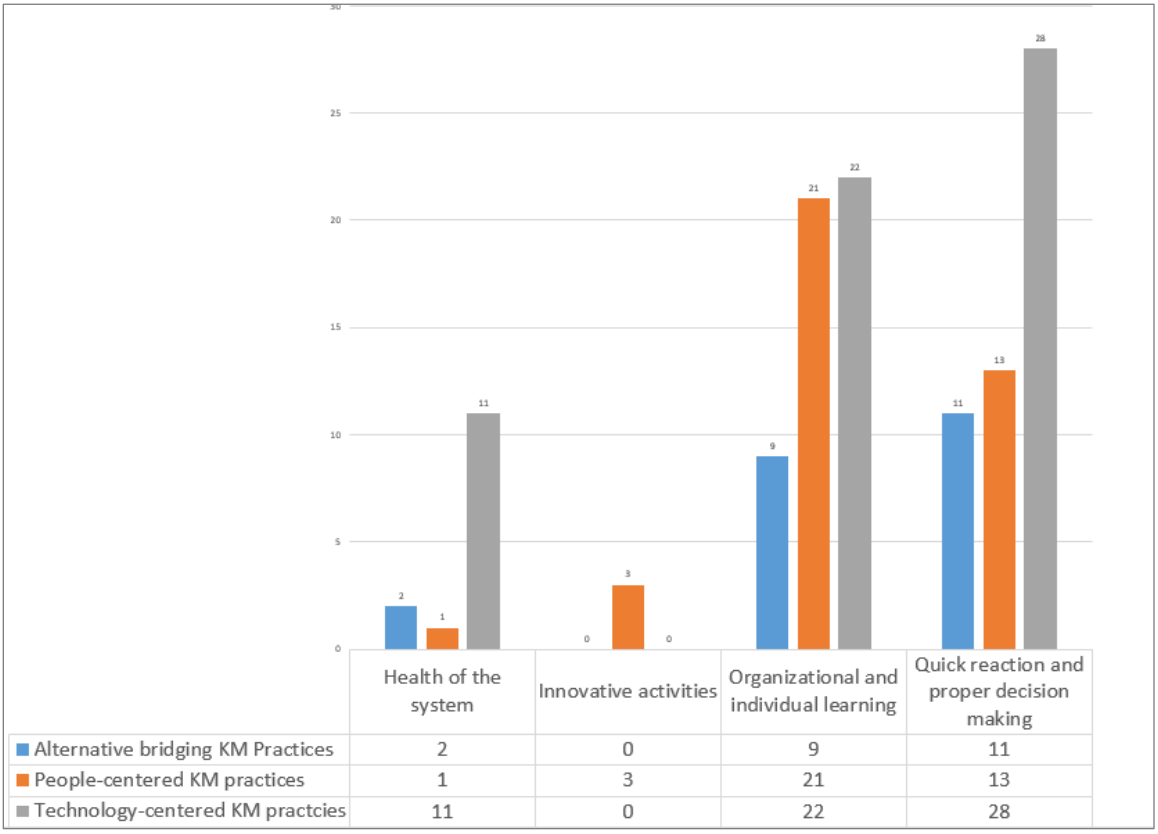
> "In design phase, we tend to have a conversation about it and where we will kind of talk about the topic and we'll propose an idea. Also, the team who might have some knowledge there might be able to say, oh, there's a risk that I can identify." (DashCo.P05)

> "collaborative activity, it gives you the chance to improve your skills gives you the chance to learn more and there is a lot of innovation." (DashC0.P04)

DevOps teams frequently express keywords that mainly focus on the health of the system and availability of the product. We summarize these first-order keywords as *health of the system* (second-order code) that constitute the DevOps performance. There is a consistent pattern in all of DevOps responses in this study that KM practices specifically technology-based KM practices increase their ability to guarantee the health of the system. For instance, as discussed in previous section, monitoring tools that convert explicit knowledge of different sources into new sets of explicit knowledge enable DevOps teams to keep track of product availability and eliminate down-time.

Thus, the above second-order codes (i.e., *Individual and organizational learning, quick reaction and proper decision making, innovative activities,* and *health of the system)* shape the aggregate dimension as DevOps performance in our data structure. (see section 3.3 for detailed data analysis process).

The code co-occurrence analysis from Dedoose in Figure 16 shows the pattern by which DevOps teams mentioned *DevOps KM practices* with *DevOps performance*:



	Health of the system	Innovative activities	Organizational and individual learning	Quick reaction and proper decision making
Alternative bridging KM Practices	2	0	9	11
People-centered KM practices	1	3	21	13
Technology-centered KM practcies	11	0	22	28

Figure 16: Codes Co-occurrence: DevOps KM Practices & DevOps Performance Outcomes

For example, DevOps personnel discussed how technology-centered KM practices such as monitoring tools allow them to support the availability and health of the system. Figure 16 simply shows these relations and that how for instance people-centered KM practices together with technology-centered KM practices paly significant role in individual and organizational learning in DevOps.

Accordingly, the following proposition can be stated:

P5:

DevOps KM practices are closely linked to achieving positive DevOps performance outcomes.

Figure 16, the analysis of DevOps KM practices and DevOps performance, leads us to the last proposition:

P6:

Technology-centered KM Practices play a relatively more important role in DevOps outcomes in terms of health of the system as well as quick reaction & proper decision making.

4.3 Summary of Findings and Emergent Theoretical Model

Following box contains six propositions discussed in result section.

P1: DevOps teams that recognized KM significance frequently engaged in KM practices. The perception of KM importance helps motivate DevOps' future KM practices.

P2: DevOps teams that recognized KM significance frequently engaged in people-centered KM practices.

P3: DevOps teams that recognized KM significance frequently engaged in technology-centered KM practices.

P4: DevOps teams that recognized KM significance frequently engaged in alternative bridging KM practices.

P5: DevOps KM practices are closely linked to achieving positive DevOps performance outcomes.

P6. Technology-centered KM Practices play a relatively more important role in DevOps outcomes in terms of health of the system and quick reaction & proper decision making.

The summary of the above-mentioned DevOps knowledge management practices is presented in Figure 17, the emergent theoretical model of this study, that shows the main components of our research question (i.e., DevOps KM practices and DevOps Performance) and their assumed interrelationships. It presents a summary of the theoretical relationships unearthed in this study. More specifically, Figure 17 depicts how the DevOps teams' perception of KM practices leads DevOps teams to engage in DevOps KM practices that are characterized by people-centered KM practices, technology-centered KM practices, and alternative bridging KM practices.

Moreover, it shows how DevOps KM practices drive DevOps performance in terms of individual and organizational learning, proper decision making, innovative activities, and health of system. This figure is the complete and revised version of the initial framework we have discussed earlier in section 2.4.

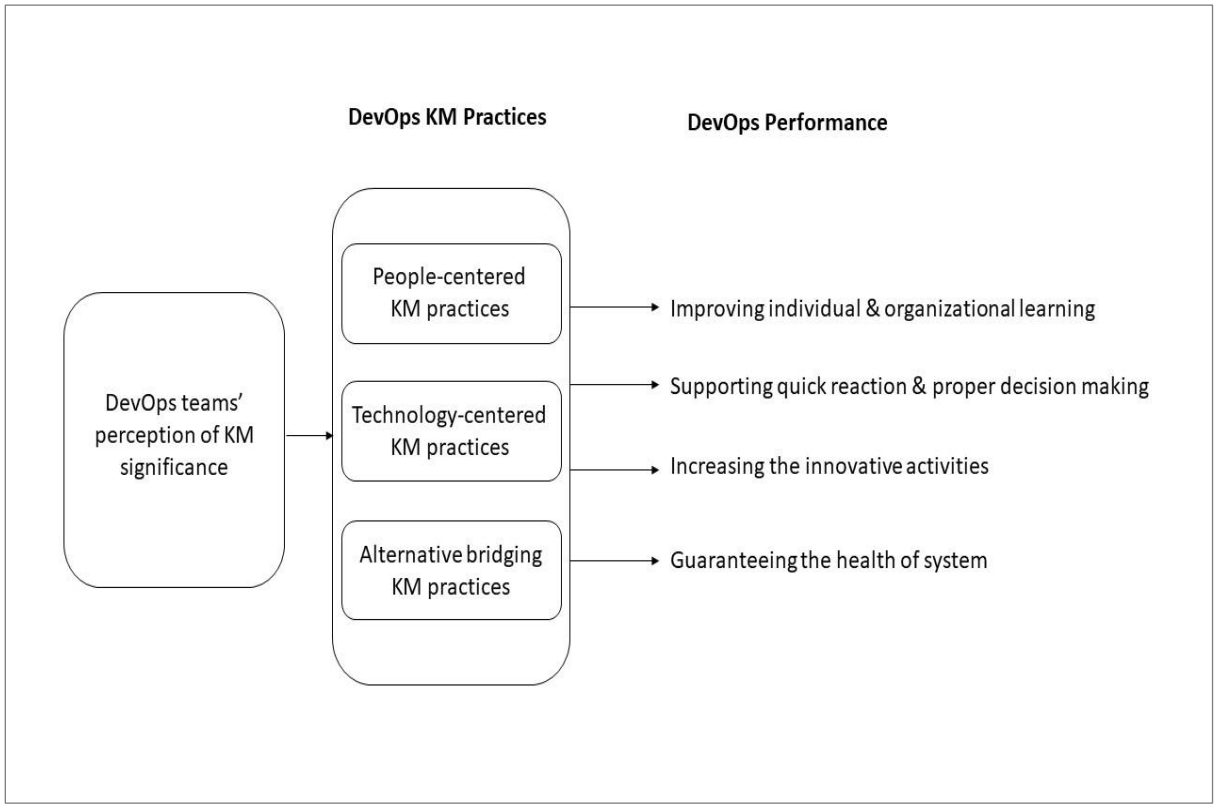


Figure 17: Enablers and Characteristics of KM Practices in DevOps & the Role of KM Practices in DevOps Performance Outcomes

Table 8 presents the case evidence for each element of the final model from one exemplary case (FinCo.).

Table 8: Exemplary evidence of themes from case FinCo.

Theme	Summary of key evidence	Quotes from participants
Perception of KM significance	Awareness of negative consequences of weak KM practices, challenges	"the biggest challenge we have, it's like without a KM platform that glue things together, it is very difficult for different teams to figure out what is best practice, which kind of functions and which they should be using from this tool, which construction they shouldn't use"
People-centered KM practices	Exploring Dev and Ops Tacit knowledge voluntary practice, Having intensive and daily interactions of Dev and Ops teams	"we have a mechanism called Guild and Chant program. So, the view is basically a group of people who have common interests on certain technology or certain topics; For example, there's a guild called Machine Learning, so the guild is cross-functional cross-teams. They would have regular meetings. They would have meet ups"
Technology-centered KM practices	Converting knowledge via communication/documentation tools, Tracking and operating tasks in teams through collaborative tools	"I would say the email system, or the instant message system is probably the most important channel. We have Confluence…People use JIRA for knowledge sharing and communication tools as well. So, another very important documentation sharing kind of knowledge sharing is Google sheet, where we use a lot of like Google doc."
Alternative Bridging KM practices	Using KM features embedded in Dev and Ops technologies	"documentation and the communication within tool, it is very effective because it's part of a workflow we're trying to do as much as possible"
DevOps Performance	Individual learning, Innovative activities	"it gives you the chance to improve your skills, gives you the chance to learn more and there is a lot of innovation."

5 Discussion and Conclusion

This section first discusses the findings with respect to the two research questions. Then we will highlight contributions to the research literature. Finally, this section concludes with the limitations, implications for the research, and implications for the practice.

5.1 Discussion of Results

This study extends KM to the context of DevOps teams to answer the following research questions 1) What are the enablers of KM practices in DevOps teams? 2) What are the distinctive characteristics of KM practices that underpin positive DevOps performance outcomes? Using inductive research design and qualitative research methods, this study followed a multiple case study approach and collected and analyzed data from 9 in-depth interviews of DevOps professionals to investigate how DevOps teams acquire, capture, share and apply knowledge. In our findings, we presented an emergent theoretical model that links KM practices to DevOps performance outcomes, and also formulated propositions to summarize our findings.

What are the enablers of KM practices in DevOps teams? (RQ1)

Most participants acknowledged that DevOps teams' perception of KM significance is a key driver for engaging in KM practices. DevOps personnel vary their description of perception of KM significance depending on whether they put their emphasis on either the awareness of KM challenges or KM best practices. The former refers to the negative consequences of weak KM practices in DevOps teams that enable them to realize the importance of KM practices. DevOps teams that faced perilous times due to lack of structured documentation practices put their effort to increasingly store and update code changes, checklists, tools' technical manuals, and other documents. In addition to awareness of KM challenges, the awareness of best KM practices can also encourage DevOps teams to improve knowledge sharing activities. DevOps leaders realized the positive consequences of best KM practices by scanning other organizations and DevOps entities that successfully implement KM ecosystems in their teams. Thus, DevOps adopt new ways of knowledge-related standards and practices tested by pioneer DevOps organizations.

The result of our study signifies the importance of both awareness of KM challenges and awareness of best practices when describing the perception of the importance of KM. Referring to the first research question of this study, DevOps teams realize the importance of KM practices that

move them toward practices by which knowledge can be managed. Figure 17 gives a summary of findings that link the DevOps teams' perception of KM significance to DevOps KM practices.

What are the distinctive characteristics of KM practices that underpin positive DevOps performance outcomes? (RQ2)

Drawing upon a practice perspective of KM that integrates two components of practice (possession and knowing) we determine the scope of KM in this study and explain how DevOps teams along with the advanced elements of DevOps technological resources are key components of this view of KM that constitute the practice (s). Accordingly, DevOps teams participate in a web of practices that we called *DevOps KM* practices characterized by people-centered, technology-centered, and alternative bridging KM practices.

People-centered KM practices refer to face-to-face collaborative activities that bring Dev and Ops teams together regularly in common place with the purpose of exploring and sharing tacit and explicit knowledge between DevOps teams' members. In addition, DevOps KM practices also regularly utilize technologies that enable conversion of an individual's tacit knowledge into explicit knowledge, explicit knowledge into new sets of explicit knowledge, and explicit knowledge into a tacit knowledge of individuals. These concepts link our data to previous research that identified two main KM approaches as personalization and codification (Hansen et al., 1999). Data suggests that codification which is a technological infrastructure for KM is more common in DevOps. Alternative and bridging KM practices describe how DevOps entities can support continual and reliable KM. Table 9 summarizes some of the DevOps KM practices identified by interviews. Appendix 7.11 presents a complete version of this table.

Table 9: DevOps KM practices: characteristics and examples from data

Characteristics of KM Practices	**Examples of DevOps KM Practices**
People-centered KM practices	• In-person pair programming, mentoring • Collaboration between mentors and apprentices • Daily scrum meeting, Sync Up meetings, Interest Group program • Less formal communications, Lighting talks • Exploring Dev and Ops tacit knowledge

Technology-Centered	• communication/documentation/collaborative technologies • Automating the knowledge-related practices • Architecture Decision making Records (ADR) online forms • Instant Message System • Monitoring technologies • Learning by doing via technologies such as online courses • Developers self-service environment and deployments. • Using shared log files and joint platform
Alternative bridging KM practices	• KM features embedded in Dev and Ops technologies • Git Hub PR • Integrated and continual KM practices and tools • Building in-house KM platform • Automate the KM practices

In a nutshell, this study explores the DevOps KM practices and extracts those which are characterized by people-centered KM practices, technology-centered KM practices, and alternative bridging KM practices. By doing this, we learn that there is an obvious link between these DevOps KM practices and certain DevOps performances. DevOps frequently explained how engaging in above KM practices helps them to a) improve individual and organizational learning, b) support quick reaction and proper decision making, c) enhance innovative activities, and d) guaranteeing the health of the system. By linking different configurations of DevOps KM practices to DevOps performance outcomes, this study found that technology-centered KM practices are popular among DevOps teams and that these practices allow DevOps to overcome changing environment and to support reliable product. For the individual and organizational learning aspect of DevOps performance, technology-centered and people-centered KM practices go hand in hand to positively improve business performance.

5.2 Contributions

This study leads to four theoretical contributions. First, it answers a recent call for a better understanding of DevOps knowledge management practices (Nielsen et al., 2017). This study is an example of how knowledge management theories might usefully be leveraged to advance the DevOps research by exploring KM practices and tools in DevOps teams. Given that the potential of practice perspective in KM discipline is well known (Mueller et al., 2011; Orlikowski, 2002; Tavakoli & Schlagwein, 2016), this study adopts practice perspective to contribute to DevOps

literature. The insight of this paper, thus, could be used to explain the enablers of KM practices in DevOps teams. Equally important, the findings of this study illustrate how DevOps KM practices drive DevOps performance. More importantly, this study sheds light on DevOps literature by introducing four DevOps performance outcomes which have not been explored in much detail in previous literature.

Second, although early attempt theorized that leaders' perception of KM is relevant for organizational effectiveness (Lakshman, 2007), the role of DevOps teams' perception that motivates engaging in KM practices across DevOps teams is new. The findings reveal that the source of DevOps teams' perception of KM significance is the extent to which they experienced KM *best practices* as well as *challenges*. This could move KM literature forward and could make key contributions to examine and understand the nature of KM.

Third, given that the knowledge combination process consists of both "collection" and "break-down" of the business concepts (Nonaka et al., 2000), we believe that DevOps context is a good example of how knowledge on one hand could be collected through continuous feedback loop, continuous monitoring, and continuous planning between developers and operations. On the other hand, DevOps teams could prioritize the responsibilities to drive a decision and changes to the system by breaking-down such comprehensive and collective knowledge. This can move both DevOps and KM literatures forward and make key contribution to understanding of real nature of DevOps.

Forth, the implications of our model align with the claim by sociotechnical literature that agile practices refer to a socio-technical approach and that it focuses on cross-functional, collaborative and team-oriented development rather than rigid software development (Davenport, 2010; Jabbari et al., 2016). Following recent research, this study focuses on DevOps teams as an example of agile organizational entities and introduces DevOps KM practices by integrating people-centered KM practices and technology-centered KM practices.

In the same vein, identifying *alternative and bridging KM practices* as DevOps KM practices contribute to both DevOps and KM research by distinguishing between more and less agile organizational entities. Previous research directs scholarly attention to further investigate agile work methods and their implications for KM (Cao, Mohan, Xu, & Ramesh, 2009). This study adds to the growing volume of KM literature that either identifies evidence of a shift towards the

hybrid agile-traditional knowledge sharing techniques (Cram & Marabelli, 2018) or addresses the differences between traditional KM and agile KM (Davenport, 2010). The insight of this paper concerning alternative and bridging KM practices could be used to shed light on agile and sociotechnical literature.

5.3 Implications for Research

The insight of this paper could be used to refine our understanding of DevOps KM. For instance, different configurations of KM practices (i.e., people-centered, technology-centered, and alternative bridging KM practices) might help explain various aspects of DevOps performance. Applied to our model, it might be worthwhile to analyze not only which KM practices are needed to improve certain DevOps performance but also which KM practices are needed so that DevOps teams are able to bridge the Dev and Ops principles and performances.

Regarding implications for alternative bridging KM practices, evidence suggests that without certain practices (e.g. integrated KM platform and KM features embedded within DevOps technologies), DevOps teams fail to support continuous improvement of the product. Future research should try to further investigate such alternative bridging KM practices in more detail across different types of organizations from SMEs to large-sized companies.

One of the limitations of this research is the small sample size of 9 participants recruited from three case organizations. Future research could target a broader cross-section of organizations and participants in different roles. Findings of this study are also limited to DevOps entities that share some similar characteristics in terms of products and clients. All three cases in this study provide enterprise applications and analytics software for business clients. Therefore, this study did not systematically examine how different types of business models (e.g. B2C vs. B2B) and different types of products (e.g. standardized vs. customized software solutions) might affect KM practices in DevOps. Further research is needed to explore DevOps KM practices across a variety of industry sectors, business models, and product types.

In addition to above, there are many micro and macro variables such as DevOps culture, organization size, cost, IT infrastructure, to name a few, that could affect DevOps KM practices and hence DevOps performance. For instance, considering that all companies in this study were based in North America and that the national culture might affect software development practices

(Ruhi & Al-Mohsen, 2015), this study does not provide any insight on the impact of international cultural factors or outsourced development in DevOps contexts. To better understand DevOps KM, future studies could explore such cultural theoretical constructs in the study of DevOps KM practices. Another interesting path for future research could be to study the evolution of KM practices alongside DevOps maturity levels. A longitudinal investigation such as this was beyond the scope of the current study.

Lastly, it should be noted the propositions outlined in the findings of this study are merely based on analytical observations, and these may not be generalizable across DevOps contexts. Future research could aim to empirically test some of these propositions and refine our understanding of KM practices in DevOps.

5.4 Implications for Practice

DevOps leaders hoping to implement a useful KM ecosystem need to realize that establishing proper awareness of KM challenges and best practices among team members are an important precursor to implementing the right KM tools and processes, and institutionalizing a good KM culture in their DevOps teams. Although it is not usual to share previous challenges with new hired DevOps member, this appears to be exactly what is need in order for DevOps employees to actively engage in KM practices. Thus, companies should provide access to resources on KM (e.g. seminars, webinars, case studies, workshops, etc.) that show that KM is an important part of DevOps. Managers who are exposed to best practices of other DevOps entities can recognize opportunities better and adopt new ways of managing knowledge in their own team.

Our concept of alternative and bridging KM practices has important practical implications for DevOps leaders. Our results suggest that the more the integration between DevOps KM practices, the more the improvement of DevOps performance. The findings of this study encourage DevOps managers to encourage and help their teams to use the KM features within DevOps technologies, and simultaneously make a link of their activities into other integrated applications and software. This would facilitate future test and technical reviews, and collaboration among Dev and Ops personnel.

Another implication for DevOps is the way DevOps teams view the documentation practices. Our result suggests that the opportunity for successful documentation is lost if documentation is

treated only as a way of storing the knowledge with no plan for future use. Data suggests that DevOps should treat documentation as a product and as a result of a source code repository. This leads us to suggest that DevOps personnel and managers look at documentation practices through a different lens. They should make sure that all procedures are documented and updated in an integrated documentation platform. They can encourage their developers to take advantage of repository and communication features within DevOps tools in order to save time and store the steps while building and modifying codes.

Given that technology-centered KM practices play a relatively more important role in positive DevOps performance outcomes, companies should invest more in implementing both open-source and in-house KM platforms. Data suggests that collaborative, communication, and monitoring tools are a must for DevOps teams to be able to track and operate tasks as one team. This way managers can make a proper decision in changing environment of DevOps and guarantee the health of the product.

Findings of this study also can be used by DevOps managers to explore innovative ideas of individuals in their team. Our analysis suggests that companies can benefit by fostering different voluntary knowledge sharing practices without specific boundaries. Short talks, stand-ups, intensive and daily interactions of Dev and Ops are examples of KM practices by which companies can gain from their own human resources to capture new ideas and improve the individual and organizational learning at lower cost.

To conclude, this study is an example of how KM practices can support the fast-paced nature of DevOps entities. Despite many KM challenges confronting DevOps, this study suggests that some DevOps teams have already established a web of KM practices and successfully provided a roadmap for others to follow. This study clearly identifies the positive role of KM practices in DevOps performance outcomes. We hope that our study provides a modest foundation to advance research and practice in DevOps KM in the future.

6 References

Alavi, & Leidner, D. E. (2001). Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS Quarterly: Management Information Systems*, *25*(1), 107–136.

Alavi, M., Kayworth, T. R., & Leidner, D. E. (2005). An empirical examination of the influence of organizational culture on knowledge management practices. *Journal of Management Information Systems*, *22*(3), 191–224. https://doi.org/10.2753/MIS0742-1222220307

Alavi, M., & Leidner, D. (1999). Knowledge Management Systems: Issues, Challenges, and Benefits. *Communications of the Association for Information Systems*, *1*, 7. https://doi.org/10.17705/1cais.00107

Anand, A., Kant, R., & Singh, M. D. (2013). Knowledge sharing in SMEs: modelling the barriers. *International Journal of Management and Enterprise Development*, *12*(4/5/6), 385. https://doi.org/10.1504/IJMED.2013.056441

Atlassian. (n.d.). Scrum - what it is, how it works, and why it's awesome. Retrieved April 1, 2020, from https://www.atlassian.com/agile/scrum

ATTLASIAN. (n.d.). Atlassian. Retrieved May 25, 2020, from https://www.atlassian.com/devops/maturity-model

ATTLASIAN. (2018). What is DevOps? | Atlassian. Retrieved May 11, 2020, from https://www.atlassian.com/devops

Balijepally, V., Mahapatra, R., & Nerur, S. P. (2006). Assessing Personality Profiles of Software Developers in Agile Development Teams. *Communications of the Association for Information Systems*, *18*. https://doi.org/10.17705/1cais.01804

Baltrusch, R. (2001). Exploring organisational learning in virtual forms of organisation. *Proceedings of the Hawaii International Conference on System Sciences*, 106. https://doi.org/10.1109/HICSS.2001.926494

Bang, S. K., Chung, S., Choh, Y., & Dupuis, M. (2013). A grounded theory analysis of modern web applications: Knowledge, skills, and abilities for DevOps. In *RIIT 2013 - Proceedings of the 2nd Annual Conference on Research in Information Technology* (pp. 61–62). https://doi.org/10.1145/2512209.2512229

Bell DeTienne, K., Dyer, G., Hoopes, C., & Harris, S. (2004). Toward a Model of Effective Knowledge Management and Directions for Future Research: Culture, Leadership, and CKOs. *Journal of Leadership & Organizational Studies*, *10*(4), 26–43. https://doi.org/10.1177/107179190401000403

Bhatt, G. D. (2001). Knowledge management in organizations: Examining the interaction between technologies, techniques, and people. *Journal of Knowledge Management*, *5*(1), 68–75. https://doi.org/10.1108/13673270110384419

Bhojaraju, G. (2005). Knowledge management: Why do we need it for corporates. *Malaysian Journal of Library and Information Science*, *10*(2), 37–50. https://doi.org/10.2139/ssrn.3375572

Cabrera, A., & Cabrera, E. F. (2002). Knowledge-Sharing Dilemmas. *Organization Studies*, *23*(5), 687–710. https://doi.org/10.1177/0170840602235001

Cao, L., Mohan, K., Xu, P., & Ramesh, B. (2009). European Journal of Information Systems A framework for adapting agile development methodologies A framework for adapting agile development methodologies. *European Journal of Information Systems*, *18*, 332–343. https://doi.org/10.1057/ejis.2009.26

Cecez-Kecmanovic, D., Galliers, R. D., Henfridsson, O., Newell, S., & Vidgen, R. (2014). The sociomateriality of information systems: Current status, future directions. *MIS Quarterly: Management Information Systems*, *38*(3), 809–830. https://doi.org/10.25300/MISQ/2014/38:3.3

Cerchione, R., Esposito, E., & Spadaro, M. R. (2016, May 1). A literature review on knowledge management in SMEs. *Knowledge Management Research and Practice*. Palgrave Macmillan Ltd. https://doi.org/10.1057/kmrp.2015.12

Chandra, Y., & Shang, L. (2019). *Qualitative research using R : a systematic approach*.

Charmaz, K. (2006). Constructing Grounded Theory: A Practical Guide through Qualitative Analysis - Kathy Charmaz - Google Books. Retrieved December 3, 2019, from https://books.google.ca/books?hl=en&lr=&id=2ThdBAAAQBAJ&oi=fnd&pg=PP1&dq=Coding+in+grounded-theory+practice.'+In+Charmaz,+K.+2006.+Constructing+grounded+theory:+A+practical+guide+through+qualitative+analysis.+&ots=fZpW7KqyyT&sig=I3SDvemM9gm7J9Ri2OMeImdwq4

Chen, C.-W., Chang, M.-L., Tseng, C.-P., Chen, B.-C., & Chang, Y. Y.-C. (2013). Retracted: Critical Human Factor Evaluation of Knowledge Sharing Intention in Taiwanese Enterprises. *Human Factors and Ergonomics in Manufacturing & Service Industries*, *23*(2), 95–106. https://doi.org/10.1002/hfm.20300

Chua, A. (2004). Knowledge management system architecture: A bridge between KM consultants and technologists. *International Journal of Information Management*, *24*(1), 87–98. https://doi.org/10.1016/j.ijinfomgt.2003.10.003

Cook, S. D. N., & Seely Brown, J. (1999). Bridging Epistemologies: The Generative Dance Between Organizational Knowledge and Organizational Knowing. https://doi.org/10.1287/orsc.10.4.381

Corbin, R. D., Dunbar, C. B., & Zhu, Q. (2007). A three-tier knowledge management scheme for software engineering support and innovation. *Journal of Systems and Software*, *80*(9), 1494–1505. https://doi.org/10.1016/j.jss.2007.01.013

Cram, W. A., & Marabelli, M. (2018). Have your cake and eat it too? Simultaneously pursuing the knowledge-sharing benefits of agile and traditional development approaches. *Information and Management*, *55*(3), 322–339. https://doi.org/10.1016/j.im.2017.08.005

Creswell, J. W., & Cuevas Shaw Karen Greene Denise Santoyo Jamie Robinson, L. (2018). *Second Edition QUALITATIVE INQUIRY& RESEARCH DESIGN Choosing Among Five Approaches Library of Congress Cataloging-in-Publication Data* (4th ed.).

Cupiał, M., Szeląg-Sikora, A., Sikora, J., Rorat, J., & Niemiec, M. (2018). Information technology tools in corporate knowledge management. *Ekonomia i Prawo, 17*(1), 5. https://doi.org/10.12775/eip.2018.001

Davenport, T. H. (2010). Process Management for Knowledge Work. In *Handbook on Business Process Management 1* (pp. 17–35). https://doi.org/10.1007/978-3-642-00416-2_2

Davis, J., & Daniels, R. (2016). Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling ... - Jennifer Davis, Ryn Daniels - Google Books. Retrieved November 18, 2019, from https://books.google.ca/books?id=6e5FDAAAQBAJ&printsec=frontcover&dq=Effective+DevOps&hl=en&sa=X&ved=0ahUKEwjuroXmoPTlAhXkT98KHZoLAbsQ6AEIKTAA#v=onepage&q=Effective DevOps&f=false

Delen, D., Zaim, H., Kuzey, C., & Zaim, S. (2013). A comparative analysis of machine learning systems for measuring the impact of knowledge management practices. *Decision Support Systems, 54*(2), 1150–1160. https://doi.org/10.1016/j.dss.2012.10.040

Detlor, B., Ruhi, U., Turel, O., Bergeron, P., Choo, C. W., Heaton, L., & Paquette, S. (2006). The Effect of Knowledge Management Context on Knowledge Management Practices: An Empirical Investigation. *The Electronic Journal of Knowledge Management*, *4*, 117–128. Retrieved from www.ejkm.com

DORA. (2018). State of DevOps, 1–74. Retrieved from https://services.google.com/fh/files/misc/state-of-devops-2019.pdf

Egbu, C. O., Hari, S., & Renukappa, S. H. (2005). Knowledge management for sustainable competitiveness in small and medium surveying practices. *Structural Survey*, *23*(1), 7–21. https://doi.org/10.1108/02630800510586871

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *The Academy of Management Review*, *14*(4), 532–550.

Erich, F. (2019). Devops is simply interaction between development and operations. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11350 LNCS*, 89–99. https://doi.org/10.1007/978-3-030-06019-0_7

Erich, F., Amrit, C., & Daneva, M. (2014a). Cooperation between information system development and operations: A literature review. *International Symposium on Empirical Software Engineering and Measurement*, 1–1. https://doi.org/10.1145/2652524.2652598

Erich, F., Amrit, C., & Daneva, M. (2014b). Report: DevOps Literature Review ENGINEERING QUALITY REQUIREMENTS IN LARGE SCALE DISTRIBUTED AGILE ENVIRONMENT View project Information Security Investments View project Report: DevOps Literature Review. https://doi.org/10.13140/2.1.5125.1201

Evangelista, P., Esposito, E., Lauro, V., & Raffa, M. (2010). The Adoption of Knowledge Management Systems in Small Firms. *Electronic Journal of Knowledge Management*, *8*, 33–42.

Fruhling, A., & De Vreede, G.-J. (2006). Field experiences with eXtreme Programming:

Developing an emergency response system. *Journal of Management Information Systems*, *22*(4), 39–68. https://doi.org/10.2753/MIS0742-1222220403

Gholami, M. H., Asli, M. N., Nazari-Shirkouhi, S., & Noruzy, A. (2013). *Investigating the Influence of Knowledge Management Practices on Organizational Performance: An Empirical Study*.

Gioia, D. A., Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2012). Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. Organizational Research Methods. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.887.8586

Guerriero, M., Ciavotta, M., Gibilisco, G. P., & Ardagna, D. (2015). SPACE4Cloud: A DevOps environment for multi-cloud applications. In *1st International Workshop on Quality-Aware DevOps, QUDOS 2015 - Proceedings* (pp. 29–30). https://doi.org/10.1145/2804371.2804378

Gupta, B., Iyer, L. S., Aronson, J. E., Gupta, B., Iyer, L. S., & Aronson, J. E. (2000). Knowledge Management: Practices and Challenges. *Industrial Management and Data Systems*, *100*(1), 17–21. https://doi.org/10.1108/02635570010273018

Hansen, M. T., Nohria, N., & Tierney, T. (1999). What's your strategy for managing knowledge? *Harvard Business Review*, *77*(2). Retrieved from www.hbr.org

Hekkala, R., Stein, M. K., & Rossi, M. (2014). "Omega-team is moving to another premise over my dead body..." Power as discursive-material practice in an is project. In *35th International Conference on Information Systems "Building a Better World Through Information Systems", ICIS 2014*.

Hemon, A., Lyonnet, B., Rowe, F., & Fitzgerald, B. (2019). From Agile to DevOps: Smart Skills and Collaborations. *Information Systems Frontiers*. https://doi.org/10.1007/s10796-019-09905-1

Hennink, M., Hutter, I., & Bailey, A. (2011). *Qualitative research methods.*

Humble, J., & Molesky, J. (2011). Why enterprises must adopt devops to enable continuous delivery. *Cutter IT Journal*, *24*(8), 6–12.

Jabbari, R., Ali, N. Bin, Petersen, K., & Tanveer, B. (2016). What is DevOps? A systematic mapping study on definitions and practices. In *ACM International Conference Proceeding Series* (Vol. 24-May-201). Association for Computing Machinery. https://doi.org/10.1145/2962695.2962707

Jarrahi, M. H., & Sawyer, S. (2013). Social Technologies, Informal Knowledge Practices, and the Enterprise. *Journal of Organizational Computing and Electronic Commerce*, *23*(1–2), 110–137. https://doi.org/10.1080/10919392.2013.748613

Joshi, Y., Parmer, S., & Chandrawat, S. S. (2012). *Knowledge sharing in organizations: modeling the barriers, an interpretive structural modeling approach.*

Kankanhalli, A., & Tan, B. C. Y. (2004). A review of metrics for knowledge management systems and knowledge management initiatives. In *Proceedings of the Hawaii International Conference on System Sciences* (Vol. 37, pp. 3717–3724).

https://doi.org/10.1109/hicss.2004.1265574

Kim, G., Humble, J., Debois, P., & John, W. (2016). The DevOps Handbook:: How to Create World-Class Agility, Reliability, and ... - Gene Kim, Jez Humble, Patrick Debois, John Willis - Google Books. Retrieved November 22, 2019, from https://books.google.ca/books?hl=en&lr=&id=ui8hDgAAQBAJ&oi=fnd&pg=PT7&dq=the+devops+handbook+how+to+create+world-class+agility+reliability+and+security+in+technology&ots=WudggfKwXQ&sig=lCLc0LlSV4MLmANu66OY0LKll38#v=onepage&q=the devops handbook how to cre

King, W. R., Qureshi, S., Kamal, M., & Keen, P. (2009). *Knowledge Management and Organizational Learning*. Retrieved from http://digitalcommons.unomaha.edu/facultybookshttp://digitalcommons.unomaha.edu/facultybooks/302

Korstjens, I., & Moser, A. (2018). Series: Practical guidance to qualitative research. Part 4: Trustworthiness and publishing. *European Journal of General Practice*, *24*(1), 120–124. https://doi.org/10.1080/13814788.2017.1375092

Kubátová, J. (2013a). Effective knowledge sharing through social technologies. In *Proceedings of the European Conference on Knowledge Management, ECKM* (Vol. 1, pp. 372–379). Retrieved from https://search.proquest.com/docview/1860698149?pq-origsite=gscholar

Kubátová, J. (2013b). Effective Knowledge Sharing Through Social Technologies - ProQuest. Retrieved December 3, 2019, from https://search.proquest.com/docview/1860698149?pq-origsite=gscholar

Lakshman, C. (2007). Organizational knowledge leadership: A grounded theory approach. *Leadership and Organization Development Journal*, *28*(1), 51–75. https://doi.org/10.1108/01437730710718245

Liao, S.-H. (2009). *The Relationship among Knowledge Management, Organizational Learning, and Organizational Performance*. *International Journal of Business and Management* (Vol. 4).

Lin, H. F. (2014). Contextual factors affecting knowledge management diffusion in SMEs. *Industrial Management and Data Systems*, *114*(9), 1415–1437. https://doi.org/10.1108/IMDS-08-2014-0232

López-Nicolás, C., & Meroño-Cerdán, Á. L. (2011). Strategic knowledge management, innovation and performance. *International Journal of Information Management*, *31*(6), 502–509. https://doi.org/10.1016/j.ijinfomgt.2011.02.003

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015). *Dimensions of devOps*. *Lecture Notes in Business Information Processing* (Vol. 212). https://doi.org/10.1007/978-3-319-18612-2_19

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2016). An Exploratory Study of DevOps Extending the Dimensions of DevOps with Practices. Retrieved from http://www.google.com

Marling, L. (2004). Knowledge management: People, process, technology. Retrieved December 3, 2019, from https://www.researchgate.net/publication/294231218_Knowledge_management_People_pro

cess_technology

Marwick, A. D. (2001). Knowledge management technology. *IBM Systems Journal*, *40*(4), 814–830. https://doi.org/10.1147/sj.404.0814

McIver, D., Lengnick-Hall, C. A., Lengnick-Hall, M. L., & Ramachandran, I. (2013). Understanding work and knowledge management from a knowledgein-practice perspective. *Academy of Management Review*, *38*(4), 597–620. https://doi.org/10.5465/amr.2011.0266

Miles, M., Huberman, A. M., & Saldaña, J. (1994). Qualitative Data Analysis A Methods Sourcebook Edition.

Miłosz, M. (2010). *Critical success factors and barriers to the implementation of knowledge management systems in Polish SMEs*. Retrieved from https://www.researchgate.net/publication/235675179

Moser, A., & Korstjens, I. (2018). Series: Practical guidance to qualitative research. Part 3: Sampling, data collection and analysis. *European Journal of General Practice*, *24*(1), 9–18. https://doi.org/10.1080/13814788.2017.1375091

Mueller, J., Hutter, K., Fueller, J., & Matzler, K. (2011). Virtual worlds as knowledge management platform - a practice-perspective. *Information Systems Journal*, *21*(6), 479–501. https://doi.org/10.1111/j.1365-2575.2010.00366.x

Murphy, N. R., Beyer, B., Jones, C., & Petoff, J. (2016). Site Reliability Engineering: How Google Runs Production Systems - Betsy Beyer, Chris Jones, Jennifer Petoff, Niall Richard Murphy - Google Books. Retrieved November 18, 2019, from https://books.google.ca/books?hl=en&lr=&id=_4rPCwAAQBAJ&oi=fnd&pg=PP1&dq=site+reliability+engineering+google&ots=pzkJ5GNQSn&sig=M8cpwF0UD1FOX6lZEEm6FxA2SyE&redir_esc=y#v=onepage&q=site reliability engineering google&f=false

Nicolau de França, B. B., Jeronimo, H., & Travassos, G. H. (2016). Characterizing DevOps by hearing multiple voices. In *ACM International Conference Proceeding Series* (pp. 53–62). https://doi.org/10.1145/2973839.2973845

Nielsen, P. A., Winkler, T. J., & Nørbjerg, J. (2017). Closing the IT development-operations gap: The devops knowledge sharing framework. In *CEUR Workshop Proceedings* (Vol. 1898).

Nonaka, I. (1994). A Dynamic Theory of Organizational Knowledge Creation. *Organization Science*, *5*(1), 14–37. https://doi.org/10.1287/orsc.5.1.14

Nonaka, I., Toyama, R., & Konno, N. (2000). SECI, Ba and Leadership: A Unified Model of Dynamic Knowledge Creation. *Long Range Planning*, *33*(1), 5–34. https://doi.org/10.1016/S0024-6301(99)00115-6

Nonaka, I., & von Krogh, G. (2009). Tacit knowledge and knowledge conversion: Controversy and advancement in organizational knowledge creation theory. *Organization Science*, *20*(3), 635–652. https://doi.org/10.1287/orsc.1080.0412

Nonaka, & Takeuchi. (1995). The Knowledge-Creating Company: How Japanese Companies Create the Dynamics ... - Ikujiro Nonaka, Hirotaka Takeuchi - Google Books. Retrieved from

https://books.google.ca/books?hl=en&lr=&id=tmziBwAAQBAJ&oi=fnd&pg=PA3&dq=the+knowledge-creating+company+nonaka&ots=pSbgGM4IBv&sig=OJBM1HEonmv9RNDrIzdc7wE58ek#v=onepage&q=the knowledge-creating company nonaka&f=false

Nunes, M. B., Annansingh, F., Eaglestone, B., & Wakefield, R. (2006). Knowledge management issues in knowledge-intensive SMEs. *Journal of Documentation*, *62*(1), 101–119. https://doi.org/10.1108/00220410610642075

Obeidat, B. Y., Al-Suradi, M. M., Masa'deh, R., & Tarhini, A. (2016). The impact of knowledge management on innovation: An empirical study on Jordanian consultancy firms. *Management Research Review*, *39*(10), 1214–1238. https://doi.org/10.1108/MRR-09-2015-0214

Orlikowski, W. J. (2002). Knowing in practice: Enacting a collective capability in distributed organizing. *Organization Science*, *13*(3), 249–273. https://doi.org/10.1287/orsc.13.3.249.2776

Oseledchik, M., Ivleva, M., & Ivlev, V. (2018). *Using Social Networks in Knowledge Management System*. atlantis-press.com*. Retrieved from https://www.atlantis-press.com/proceedings/iccese-18/25894062

Patton, M. Q. (2015). The Sociological Roots of Utilization-Focused Evaluation. *American Sociologist*, *46*(4), 457–462. https://doi.org/10.1007/s12108-015-9275-8

Ridder, H. G., Miles, M. B., Huberman, A., & Saldaña, J. (2014). Qualitative data analysis. A methods sourcebook. *Zeitschrift Fur Personalforschung*, *28*(4), 485–487. https://doi.org/10.1177/239700221402800402

Ruhi, U., & Al-Mohsen, D. (2015). Enterprise 2.0 Technologies for Knowledge Management: Exploring Cultural, Organizational & Technological Factors. *IBIMA Publishing Journal of Organizational Knowledge Management*, *2015*. https://doi.org/10.5171/2015.789394

Scarso, E. (2017). Corporate universities as knowledge management tools. *VINE Journal of Information and Knowledge Management Systems*, *47*(4), 538–554. https://doi.org/10.1108/VJIKMS-12-2016-0074

Sharma, S. (2017). *The DevOps Adoption Playbook: A guide to Adopting DevOps in a Multi-Speed IT Enterprise - Chapter 1. DevOps an overview*. Retrieved from https://books.google.ca/books?hl=en&lr=&id=sJbqDQAAQBAJ&oi=fnd&pg=PR1&dq=The+DevOps+Adoption+Playbook:+A+Guide+to+Adopting+DevOps+in+a+Multi-Speed+IT+Enterprise&ots=PK43InWxif&sig=GzrgAv7BO825DNufSbbtIkdTm3k#v=onepage&q=The DevOps Adoption Playbook%3A A G

Singh, A., Singh, K., & Sharma, N. (2014). Agile knowledge management: a survey of Indian perceptions. *Innovations in Systems and Software Engineering*, *10*(4), 297–315. https://doi.org/10.1007/s11334-014-0237-z

Smeds, J., Nybom, K., & Porres, I. (2015). *DevOps: A definition and Perceived Adoption Impediments*. *Lecture Notes in Business Information Processing* (Vol. 212). https://doi.org/10.1007/978-3-319-18612-2_14

Soto-Acosta, P., Colomo-Palacios, R., & Popa, S. (2014). Web knowledge sharing and its effect on innovation: an empirical investigation in SMEs. *Knowledge Management Research & Practice*, *12*(1), 103–113. https://doi.org/10.1057/kmrp.2013.31

Spender, J.-C. (1996). Knowledge and the Firm: An Overview Article in Strategic Management Journal. https://doi.org/10.1002/smj.4250171103

Talanquer, V. (2014). Using qualitative analysis software to facilitate qualitative data analysis. In *ACS Symposium Series* (Vol. 1166, pp. 83–95). https://doi.org/10.1021/bk-2014-1166.ch005

Tavakoli, A., & Schlagwein, D. (2016). A review of the use of practice theory in information systems research. *Pacific Asia Conference on Information Systems, PACIS 2016 - Proceedings*.

Tessem, B., & Iden, J. (2008). Cooperation between developers and operations in software engineering projects. In *Proceedings - International Conference on Software Engineering* (pp. 105–108). https://doi.org/10.1145/1370114.1370141

Vieru, D., & Rivard, S. (2008). THE DILEMMA OF INTEGRATION VERSUS AUTONOMY: KNOWLEDGE SHARING IN POST-MERGER IS DEVELOPMENT Le dilemme intégration-autonomie : partage de connaissances lors du développement de systèmes d'information en contexte d'intégration post-fusion.

Washington, S. (2016). (1) DevOps vs Site Reliability Engineering | LinkedIn. Retrieved November 25, 2019, from https://www.linkedin.com/pulse/devops-vs-site-reliability-engineering-sean-washington/

Wettinger, J., Andrikopoulos, V., & Leymann, F. (2015). Automated capturing and systematic usage of DevOps knowledge for cloud applications. In *Proceedings - 2015 IEEE International Conference on Cloud Engineering, IC2E 2015* (pp. 60–65). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/IC2E.2015.23

Wittrock, M. C. (1974). Learning as a generative process1. *Educational Psychologist*, *11*(2), 87–95. https://doi.org/10.1080/00461527409529129

Yeow, A., Chua, K., Wee, J., & Chua, A. Y. K. (2013). The peculiarities of knowledge management processes in SMEs: The case of Singapore The peculiarities of knowledge management processes in SMEs-The case of Singapore. *Article in Journal of Knowledge Management*. https://doi.org/10.1108/JKM-04-2013-0163

Yew Wong, K., & Aspinwall, E. (2004). Characterizing knowledge management in the small business environment. *Journal of Knowledge Management*, *8*(3), 44–61. https://doi.org/10.1108/13673270410541033

Yin, R. K. (2017). Case Study Research and Applications: Design and Methods - Robert K. Yin - Google Books. Retrieved December 3, 2019, from https://books.google.ca/books/about/Case_Study_Research_and_Applications.html?id=uX1ZDwAAQBAJ&printsec=frontcover&source=kp_read_button&redir_esc=y#v=onepage&q&f=false

Zheng, W., Yang, B., & McLean, G. N. (2010). Linking organizational culture, structure, strategy, and organizational effectiveness: Mediating role of knowledge management.

Journal of Business Research, *63*(7), 763–771. https://doi.org/10.1016/j.jbusres.2009.06.005

7 Appendix

7.1 Ethics Approval Letter

04/07/2019

Université d'Ottawa
Bureau d'éthique et d'intégrité de la recherche

University of Ottawa
Office of Research Ethics and Integrity

CERTIFICAT D'APPROBATION ÉTHIQUE | CERTIFICATE OF ETHICS APPROVAL

Numéro du dossier / Ethics File Number	H-06-19-4285
Titre du projet / Project Title	Exploring Knowledge Management Practices and Technologies in DevOps
Type de projet / Project Type	Thèse de maîtrise / Master's thesis
Statut du projet / Project Status	Approuvé / Approved
Date d'approbation (jj/mm/aaaa) / Approval Date (dd/mm/yyyy)	04/07/2019
Date d'expiration (jj/mm/aaaa) / Expiry Date (dd/mm/yyyy)	03/07/2020

Équipe de recherche / Research Team

Chercheur / Researcher	Affiliation	Role
Soha SOLOUKI	École de science informatique et de génie électrique / School of Electrical Engineering and Computer Science	Chercheur Principal / Principal Investigator
Umar RUHI	École de gestion Telfer / Telfer School of Management	Superviseur / Supervisor
Lysanne LESSARD	École de gestion Telfer / Telfer School of Management	Co-superviseur / Co-supervisor

Conditions spéciales ou commentaires / Special conditions or comments

550, rue Cumberland, pièce 154 Ottawa (Ontario) K1N 6N5 Canada
550 Cumberland Street, Room 154 Ottawa, Ontario K1N 6N5 Canada

613-562-5387 • 613-562-5338 • ethique@uOttawa.ca / ethics@uOttawa.ca
www.recherche.uottawa.ca/deontologie | www.recherche.uottawa.ca/ethics

04/07/2019

Université d'Ottawa

Bureau d'éthique et d'intégrité de la recherche

University of Ottawa

Office of Research Ethics and Integrity

Le Comité d'éthique de la recherche (CÉR) de l'Université d'Ottawa, opérant conformément à l'*Énoncé de politique des Trois conseils* (2014) et toutes autres lois et tous règlements applicables, a examiné et approuvé la demande d'éthique du projet de recherche ci-nommé.

L'approbation est valide pour la durée indiquée plus haut et est sujette aux conditions énumérées dans la section intitulée "Conditions Spéciales ou Commentaires". Le formulaire « Renouvellement ou Fermeture de Projet » doit être complété quatre semaines avant la date d'échéance indiquée ci-haut afin de demander un renouvellement de cette approbation éthique ou afin de fermer le dossier.

Toutes modifications apportées au projet doivent être approuvées par le CÉR avant leur mise en place, sauf si le participant doit être retiré en raison d'un danger immédiat ou s'il s'agit d'un changement ayant trait à des éléments administratifs ou logistiques du projet. Les chercheurs doivent aviser le CÉR dans les plus brefs délais de tout changement pouvant augmenter le niveau de risque aux participants ou pouvant affecter considérablement le déroulement du projet, rapporter tout évènement imprévu ou indésirable et soumettre toute nouvelle information pouvant nuire à la conduite du projet ou à la sécurité des participants.

The University of Ottawa Research Ethics Board, which operates in accordance with the *Tri-Council Policy Statement* (2014) and other applicable laws and regulations, has examined and approved the ethics application for the above-named research project.

Ethics approval is valid for the period indicated above and is subject to the conditions listed in the section entitled "Special Conditions or Comments". The "Renewal/Project Closure" form must be completedfour weeks before the above-referenced expiry date to request a renewal of this ethics approval or closure of the file.

Any changes made to the project must be approved by the REB before being implemented, except when necessary to remove participants from immediate endangerment or when the modification(s) only pertain to administrative or logistical components of the project. Investigators must also promptly alert the REB of any changes that increase the risk to participant(s), any changes that considerably affect the conduct of the project, all unanticipated and harmful events that occur, and new information that may negatively affect the conduct of the project or the safety of the participant(s).

Riana MARCOTTE
Responsable d'éthique en recherche / Protocol Officer
Pour/For **Daniel LAGAREC** Président(e) du/ Chair of the **Comité d'éthique de la recherche en sciences sociales et humanités / Social Sciences and Humanities Research Ethics Board**

550, rue Cumberland, pièce 154 Ottawa (Ontario) K1N 6N5 Canada
550 Cumberland Street, Room 154 Ottawa, Ontario K1N 6N5 Canada

613-562-5387 • 613-562-5338 • ethique@uOttawa.ca / ethics@uOttawa.ca
www.recherche.uottawa.ca/deontologie | www.recherche.uottawa.ca/ethics

7.2 Interview Questions

Theme: Organizational Knowledge Management Environment

1. How would you describe the knowledge management culture in your organization as a whole?
    - How the organization considers knowledge as a key enabler or differentiator for business performance?
    - How colleagues and departments regularly share lessons learned?
    - How does the organization focus on development and sharing of best practices?
    - What are some formal and informal opportunities for knowledge sharing?
    - How does the organization offer opportunities to improve employee knowledge capabilities through training and development?
    - How about the availability and support for tools for communication and collaboration?
    - How does the organization offer encouragement and recognition for knowledge sharing?

2. What are the strengths and weakness of current KM practices for the acquisition, conversion and application of knowledge in your organization?
    - Can the leadership do something different at the strategic level?
    - Would you recommend different or new procedures or technologies at a tactical level?

Theme: DevOps

3. What is the history of DevOps in your organization? How did it start? Who led the initiative?

4. What is the purpose behind DevOps initiatives in the organization?

5. What is the configuration of the DevOps team in your organization?
    - Who are the people involved?
    - What are their professional backgrounds or job roles?
    - Is there a separate organizational entity designated as DevOps?
    - Is DevOps used in formal job titles?

6. How do you see a difference in organizational performance under a DevOps shop as opposed to how things were being done previously, or how things are done in a non-DevOps environment?

7. What best practice frameworks or standards have been used for running DevOps in your organization?

8. Would you consider your DevOps environment to be successful? Why or Why Not?

9. What were the challenges that needed to be overcome to get to your current state of DevOps? What challenges still exist?

10. Can you please comment on the technology stack and tools that are core to DevOps in your organization?
    - What are these tools used for?
    - How were the tools procured?
    - Have they changed over time?
    - How do the current tools enable better productivity or performance for the DevOps team and the organization as a whole?

Theme: Knowledge Management Practices in DevOps

11. What types of knowledge do you think is key to the effective functioning of DevOps teams? Can you provide some examples?
    - What is role of tacit and explicit knowledge in effective DevOps units?
    - Do you think there more emphasis on certain types of knowledge – e.g. transactional, analytical, process knowledge etc.?

12. How the knowledge management culture in DevOps any different from that of the rest of the organization?
    - How is it or is it not different?
    - What different processes and technologies are needed to support the fast-paced nature of DevOps?
    - How are the requirements different to support the iterative and short cycles of work in DevOps?

13. How is knowledge acquired, converted, shared, and applied in DevOps units?
    - How is knowledge derived from external sources and customers?
    - How is knowledge shared among DevOps team members?
    - How is knowledge stored or shared for future use?
    - What are the formal or informal processes of applying knowledge to modify products or outputs from DevOps?

14. How is knowledge sharing facilitated between developers and IT operations staff?
    - How does your organization foster shared responsibility and trust among a multidisciplinary team across the two IT functions?
    - What types of in-person or face-to-face meetings are used?

- How does technology enable cross-functional knowledge sharing?
- How do you balance the needs for autonomous teams versus cross-functional collaboration in DevOps?

15. What types of dedicated knowledge management systems are currently used to facilitate your DevOps practices?
    - What types of communication and collaboration technologies are used?
    - What types of content management technologies are used?
    - What types of business intelligence technologies are used?

16. Thinking about your core DevOps technology stack or toolchain, what types of knowledge management features exist in these technologies that aid DevOps practices, and how effectively are these features used?
    - What are the knowledge management features available in the technologies used for code development, testing, release management, configuration management etc.?
    - Are these features used to their full extent, or does the DevOps team prefer to use separate tools for knowledge management purposes?

17. Can you give me an example of a positive experience or outcome that you attribute to your current DevOps knowledge management practices?
    - What happened (cause, description, outcome)?
    - How did it make you feel?
    - What do you do anything differently as a result of this experience?

18. Can you give me an example of a negative experience or outcome that you attribute to your current DevOps knowledge management practices?
    - What happened (cause, description, outcome)?
    - How did it make you feel?
    - What do you do anything differently as a result of this experience?

7.3 Call for Participation Form

Invitation to Participate in M.Sc. Thesis Research Project: Exploring the Knowledge Management Practices & Technologies in DevOps

I would like to invite you to participate in an in-person or a phone interview to help me in my graduate thesis research on *Exploring the Knowledge Management (KM) Practices and Technologies in Development and Operations (DevOps)*. I am a master's student at the University of Ottawa, and I am currently in the process of collecting data for my research project. Specifically, I am requesting your participation through an in-person or a phone interview that will take between 30 to 40 minutes of your time. The interview consists of general questions on how agile organizational entities such as DevOps teams acquire, capture, share and apply knowledge effectively.

I am hoping that the findings of my study will help explain the role of effective KM practices in enhancing agility and innovation related business performance outcomes. The research aims to draw upon experiences and opinions of key personnel involved in DevOps roles in their organizations and to identify key success factors for knowledge management in such an environment. The purpose of the study is to gain a better understanding of knowledge management practices in agile work contexts such as DevOps teams.

Please note that the interview will not elicit any personally identifiable information, and we are not seeking proprietary or company sensitive information. Hence anonymity and confidentiality are guaranteed. Your choice to participate or not to participate in this study will not be reported to your organization. Results from the analysis of interview data will only be reported in anonymized or aggregate form. It will not be possible to identify any particular individual on the basis of information included in the study results. Furthermore, the responses from the interview will be kept confidential. They will not be released back to your organization or any third party. Lastly, all interview data will be kept in a safe and secure environment, and only the researchers involved in this project will have access to this information.

I would be happy to answer any questions that you might have. You may email me. Please also feel free to contact my research project supervisor, Dr. Umar Ruhi

If you have any concerns over the ethical aspects of this research, please contact:

Office of Research Ethics and Integrity, University of Ottawa
Tabaret Hall, 550 Cumberland St., Room 154
Ottawa, ON, Canada K1N 6N5
Tel.: (613) 562-5387 | Fax.: (613) 562-5338 | Email: ethics@uottawa.ca

Thank you for your assistance.

Sincerely,

Mrs. Soha Solouki (Principal Investigator)
MSc. Candidate in E-Business Technologies
University of Ottawa

Dr. Umar Ruhi (Research Supervisor)
Associate Professor
Telfer School of Management, University of Ottawa

7.4 Consent Form

Information Sheet & Consent Form for Research Project:
Exploring Knowledge Management Practices and Technologies in DevOps

You are invited to participate in an interview conducted as part of Mrs. Soha Solouki Master's Thesis Research on Exploring Knowledge Management Practices and Technologies in DevOps under the supervision of Professor Umar Ruhi.

This information sheet and consent form provides a description of our research project and the procedures that will be followed in our study. To indicate your agreement and proceed with participation in this study, please read and sign at the bottom of the page.

Mrs. Soha Solouki (Principal Investigator)
MSc. Candidate in E-Business Technologies
University of Ottawa

Dr. Umar Ruhi (Research Supervisor)
Associate Professor

Purpose of the Research Project

The purpose of the study is to gain a better understanding of knowledge management (KM) practices in agile work contexts such as DevOps teams. Specifically, the research aims to investigate how agile organizational entities acquire, capture, share and apply knowledge effectively for improved business performance. To develop an understanding of KM practices that support agile business strategy and operations, this study targets organizations that currently use DevOps principles and practices in their IT function. Research findings from this study can potentially help explain the role of effective KM practices in enhancing agility and innovation related business performance outcomes. The research aims to draw upon experiences and opinions of key personnel involved in DevOps roles in their organizations, and to identify key success factors for knowledge management in such an environment.

The research aims to make contributions to both theory and practice concerning knowledge management practices and technologies in DevOps teams.

This research project has received ethics approval by the University of Ottawa's Research Ethics Board (<<insert REB approval number>>). If you have concerns or questions about your rights as a participant or about the way the research project is conducted, you may contact:

Office of Research Ethics and Integrity, University of Ottawa
Tabaret Hall, 550 Cumberland St., Room 154
Ottawa, ON, Canada K1N 6N5
Tel.: (613) 562-5387 | Fax.: (613) 562-5338 | Email: ethics@uottawa.ca

Data Collection Procedures

Specifically, you will be responding to open-ended questions in an interview. The interview consists of general questions related to the current state of Knowledge Management (KM) practices and their role in supporting DevOps teams within organizations. The interview will take about 30 to 40 minutes and will be audio-recorded to ensure accurate transcription and interpretation of interview responses.

You can also ask to see a copy of your responses (after interview transcription is completed). To request a copy, please send me an email Please note that the transcripts will be sent via regular email.

Anonymity and Confidentiality

Your identity will remain anonymous and the data collected will be kept confidential and treated with respect. No personally identifiable information will be solicited during the actual interview, nor will it be used in our reporting of research findings. Information entered in the consent form will be kept separate from your interview data. Research findings will be reported through use of pseudonyms based on industry category and/or job role. Furthermore, we are not seeking proprietary or company sensitive information. Rest assured, all responses will be kept confidential, and they will not be released back to your own organization or any third party.

Conservation of Data

Participants' identifying information (such as this signed consent form) will be kept in a password-protected file separate from the rest of the data collected from the interview session. The signed consent forms will be kept in a locked filing cabinet in the research supervisor's office.

Upon completion of the interviews, the interview audio files as well the transcribed text files will be stored on a password secured computers belonging to the principal investigator and/or her supervisor. To ensure security and confidentiality of data, both computers have multiple layers of security software including Authenticated Operating System Access, Firewall software, Antivirus, and Malware protection.

After the completion of the research, data will be deleted securely from the principal investigator's computer, and it will be archived in a secure digital environment at the supervisor's faculty for 5 years. After this time, the digital versions of interview data will be deleted securely. To ensure secure deletion, we will use a software utility such as File Shredder that provides file deletion and space wiping functionality. Paper copies of consent forms will be shredded using the secure shredding services offered by materials management at the research project supervisor's faculty.

Secondary Use of Data

The data gathered in this study may be used by members of the research team in subsequent research investigations exploring DevOps knowledge management practices and technologies. Any secondary use of data will be treated with the same degree of confidentiality and anonymity as in the original project. As such, any research projects requiring access to the data collected for this project will also be reviewed by a research ethics board for approval.

Potential Harms or Risks

There are no physical, psychological, or social risks expected as a result of participation in this research project.

Potential Benefits

The participation in this study will allow the participants to reflect and potentially also enhance their own understanding of knowledge management practices used in their organizations. More specifically, participants may be able to identify strengths and weaknesses of their current knowledge management capabilities for DevOps and reflect on possible improvements.

Participation

Your participation in this study is strictly voluntary. If you do not want to answer all the questions in the interview, you do not have to.

If you decide to participate, you can decide to stop at any time, even after completing the consent form, part-way through the interview, or any time after the completion of the interview. If you withdraw, you have the choice of requesting that your data be deleted and not included in the reporting of the study's findings (unless these findings are already published or presented). Even after publication, you can withdraw your data preventing its use in future research as secondary data. To withdraw from the study, simply send me an email stating that you wish to withdraw and whether or not you would like your responses to be deleted. You are also welcome to contact me or my supervisor for any additional information.

Results of the Research Project

You can request a summary of the findings from our study after the analysis of the responses has been completed. If you wish to receive a summary, please send me an email stating that you wish to receive a summary of findings.

Informed Consent

I have read the information presented above about the study being conducted by Mrs. Soha Solouki from the University of Ottawa. I have had the opportunity to ask questions about my involvement in this study and to receive any additional details I wanted to know. I understand that I may withdraw from the study at any time, if I choose to do so. I understand that my confidentiality and anonymity, as outlined above, will be guaranteed. I understand that the data collected for this project could potentially be used in future research investigations.

There are two copies of the consent form, one of which is yours to keep.

Your signature below indicates that you have received a copy of this consent form for your own records.

Your signature indicates that you consent to participate in this study.

_____	_____
Participant Signature Date	Researcher Signature Date
_____	_____
Printed Name (Participant)	Printed Name (Researcher)

7.5 DevOps Tools Definition

This table describes the tools that are widely used by DevOps mentioned in this study.

Tools	Description
Jenkins	Jenkins is a free and open source automation server. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat.
Ansible	Ansible is an open-source software provisioning, configuration management, and application-deployment tool. It runs on many Unix-like systems and can configure both Unix-like systems as well as Microsoft Windows. It includes its own declarative language to describe system configuration.
Git	Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.
Git Hub-Pull Requests	Pull requests let you tell others about changes you've pushed to a branch in a repository on GitHub. Once a pull request is opened, you can discuss and review the potential changes with collaborators and add follow-up commits before your changes are merged into the base branch.
Subversion	Apache Subversion is a software versioning and revision control system distributed as open source under the Apache License. Software developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation.
You Track	You Track is a proprietary, commercial browser-based bug tracker, issue tracking system and project management software developed by JetBrains. It focuses on query-based issue search with auto-completion, manipulating issues in batches, customizing the set of issue attributes, and creating custom workflows.
Confluence	Confluence is Atlassian's content collaboration tool used to help teams collaborate and share knowledge efficiently. In Confluence, content is created and organized using spaces, pages, and blogs. Confluence's collaboration tools allow users to write, edit, comment and get work done together within the Confluence interface.
Slack	Slack is a collaboration hub people and teams work together seamlessly. It’s designed to support the way people naturally work together, so one can collaborate with people online as efficiently as he does face-to-face.
Jira	Jira Software is part of a family of products designed to help teams of all types manage work. Originally, Jira was designed as a bug and issue tracker. But today, Jira has evolved into a work management tool for different use cases, from requirements and test case management to agile software development.
Telemetry	Telemetry is the collection of measurements or other data at remote or inaccessible points and their automatic transmission to receiving equipment for monitoring.

Microsoft team	Microsoft Teams is a unified communication and collaboration platform that combines persistent workplace chat, video meetings, file storage, and application integration.
Prometheus	Prometheus is a free software application used for event monitoring and alerting. It records real-time metrics in a time series database built using a HTTP pull model, with flexible queries and real-time alerting.
Elasticsearch	Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Elasticsearch is developed in Java.
Kibana	Kibana is an open source data visualization dashboard for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data.
Docker	Docker is a set of platforms as a service products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels.

7.6 Code Book

Code (sub-code)	Description	Example from data
DevOps teams' perception of KM significance		
Awareness of KM challenges	Apply this code for discussion about negative consequences of weak KM practices. This code refers to one that realize the lack of documentation and communication practices.	"lately we had layoff, and somebody left the team and their knowledge left the team." (DashCo.P02)
Awareness of KM best practices	Use this code for mention of positive consequences of KM practices. This code refers to acknowledgement of current KM best practices.	"We are aware of different solutions and want to adopt best practices from others, in addition to contributing our best practices to the industry." (FinCo.P01)
DevOps KM practices		
People-centered KM practices	Apply this code for DevOps KM practices by which tacit and explicit knowledge can be managed through network of people and cross-functional teams (Hansen et al., 1999) This code is about intensive face-to-face interactions of Dev and Ops teams, exploring individuals' tacit knowledge by voluntary activities, and bringing together Dev and Ops for collaborative troubleshooting.	"Somebody will investigate a new technology and then through meetings convey that information to the team." (DashCo.P02)
Technology-centered KM practices	Apply this code to DevOps practices that are put in place for the purpose of identifying and managing tacit and explicit knowledge via material resources (e.g., technological infrastructure) (Hansen et al., 1999). This code focuses on tracking and operating tasks through collaborative tools, automating the knowledge-related practices, and communication and documentation tools.	"Slack is where people live and breathe these days. It's the fastest way to communicate" (DashCo.P03)

Code (sub-code)	Description	Example from data
Alternative Bridging KM practices	Apply this code for mention of alternative and supporting KM practices/tools for DevOps when the main tools are down in order to support continuous release and development. This code refers to KM features embedded in DevOps technologies as well as integrated and in-house KM platforms.	"we have structured KM platform… knowledge information discoverability it is within the platform itself… all the tools are integrated by a platform our organization provides." (FinCo.P01)
DevOps performance		
Individual & Organizational Learning	Use this code for mention of DevOps performance outcomes which is achieved when DevOps integrate learning into daily works of DevOps. Apply this code for creation and dissemination of knowledge among individuals through collaborative activities and self-driven learning.	"A lot of training is also done in self-driven, like learning a new technology" (FinCo.P02)
Quick reaction & Proper decision making	This code is about DevOps performance that enables the fast and constant reactions at all stages of the value stream. Apply this code when Dev and Ops teams detect the problems when they are cheaper, smaller, and easier to solve.	"Real-time Monitoring. It is very important in our job. There are 5 dashboards now in the company so that every team member can see the alerts and problems on time and react immediately" (DashCo.P01)
Innovative activities	This code is about DevOps performance that enables the innovative environment. Apply this code for proposing new ideas and recognizing new opportunities in DevOps.	"collaborative activity, it gives you the chance to learn more and there is a lot of innovation." (DashC0.P04)
Health of the system	This code refers to DevOps performance that is about reliable product and continuous improvement and development. Apply this code for eliminating the downtime of system.	"we are using alert manager (monitoring tool), it's like listing those events and making sure that the servers are up and running and are healthy." (DashCo, P04)

7.7 Exemplary evidence of perception of importance of KM practices in DevOps teams

Table A7.7: exemplary evidence of perception of importance of KM practices in DevOps teams

Case	primary codes	Quotes from participants
Dash Co.	Pride of current KM activities of Dev and Ops teams, Awareness of negative consequences of weak communication practices.	"we also have across the whole development team something they call Lightening Talks and what we do is making a presentation to the entire team about what you are working on. Those work very well because you can make a document. People don't necessarily read it but if they were in the presentation, they can get some information and for more details they can go to the written document. Otherwise you might have problem to put time, go to bunch of documents and get the right information. I really enjoy these kinds of meetings to get new ideas."
	Awareness of negative consequences of weak documentation practices.	"I think because we are small team it is important that we share that information across because lately we had layoff, and somebody left the team and their knowledge left the team. You do not know what you did not know until personnel leave. So, having known we've gone through this were made my effort to increase the coverage of documentation"
	Pride of current knowledge sharing activities of Dev and Ops teams	"Mentoring, in this approach the team creates a list of the abilities and needs of the employees. Based on this database, the employees can ask questions from each other… They arrange a short meeting to talk about that topic. This approach is new and very useful for me. I could say such Mentoring is one of the strengthen of current KM practices."
	Acknowledge of current KM best practices.	"we have training budget. In order to improve the knowledge capability, the company considers a training budget for each employee per month around 100$. I think this amount of money is a lot for training courses. I have passed many Udemy online courses and the company paid for them."

Table **A7.7** (continued)

Case	primary codes	Quotes from participants
Dash Co.	Awareness of negative consequences of weak documentation practices, Pride of current documentation activities of Dev and Ops teams,	"there is no well-organized documentation tool. Some of the knowledge of projects are not visible for future use. They should be updated and so it is very time-consuming for people to do…We use Confluence for documentation. It is very interesting and simple and whenever somebody make changes into it, we will be notified by email about that change."
	Acknowledge of current knowledge sharing best practices, Awareness of negative consequences of weak communication practices.	"We are sitting all close to each other and we can all share whenever we have an issue that we have anything to deal with … customers complain about something and give us for example reports of crash or say my server is down. In our case because we need something faster and we need something to react quick face-to-face communication works well."
	Awareness of negative consequences of weak communication practices.	"One thing that's very helpful is that we tend to have a conversation about in design phase…engineers tend to build the perfect solution to whatever problem. And perfect doesn't really go that well with time frames and getting things done in a reasonable time. So understanding how much you really need of the product to be completed for it to be functional help and solve the problem versus all of the things that you could potentially build into it, this is helpful to have feedback from the all team members."
	Awareness of negative consequences of weak communication practices, Acknowledge of current knowledge sharing best practices.	"if something's happening and I don't really expect to deal with it or I just happened to be away when this customer calls in, they don't want to wait. It would be nice if there's someone else in the team at least knew kind of what was going on some time. We try to give each other context on the day to day stuff there."
	Awareness of negative consequences of weak documentation practices.	Documentation is really hard to keep up to date. And when everything is changing all the time, you can spend a lot of time updating your docs for no value because you still have to figure out whatever the next problem.
FinCo	Acknowledge of current knowledge sharing best practices.	"we see a lot of teams doing, you know, they have the same problem and they solve it in two different ways and then kind of come together and say, we saw it this way. We thought it that way. What can we take from each others to make it better and come up with a central solution. And so, we definitely see kind of the independent teams working independently and coming up with solutions and then merging them at the end."

Table **A7.7** (continued)

Case	primary codes	Quotes from participants
FinCo.	Pride of current communication activities of Dev and Ops teams, Acknowledge of current knowledge sharing best practices.	"I think our organization is really heavy in tribal knowledge where you learn things and you discover them by experimenting, by asking people...it's actually really encouraged that if you need to know something, you go into a popular chat and ask, hey, where can I go ask about this question. And they'll guide you to a more a more direct one engaged to specific people or just answer there."
	Acknowledge of current knowledge sharing best practices,	"in DevOps team specifically that it's they are encouraged by every level of management to work with development teams and work with each other to ensure that the solution that's being provided as is appropriate and is scalable to what needs to be solved… So I think it's really just management at every level and encouraging kind of that cross-pollination of knowledge between related teams."
	Acknowledge of current knowledge sharing best practices.	"I encourage my entire team to participate in the wider community to learn new technologies and to expose the rest of our team to the new technologies and practices that are going on and kind of move forward from there and stay involved in what's going on that."
	Awareness of negative consequences of weak communication practices,	"one of the most effective skill I think is going to more on the soft skills side, it's gonna be communication. A large role of a successful SRE and dev ops is convincing other engineering teams and communicating with other engineering teams that a specific choice or a specific direction is the right way to go."
	Awareness of negative consequences of weak documentation practices.	"I think the lesson learned is that treating documentation like a product, like a result, like a releasable thing, I think is the direction we need to head."
	Acknowledge of current knowledge sharing best practices,	"I also stay current on industry trends by attending both internal and external events and conferences, and we talk to teams from other technology firms with similar challenges. We are increasingly presenting at conferences too. We are aware of different solutions and want to adopt best practices from others, in addition to contributing our best practices to the industry."
	Pride of current communication activities of Dev and Ops teams, Awareness of negative consequences of weak documentation practices.	"clear documentation is also part of my organization where my team will provide training material both in documentation platform as well as in-person training to make sure our engineers can get a necessary training to do their job."

Table **A7.7** (continued)

Case	primary codes	Quotes from participants
FinCo.	Awareness of negative consequences of weak communication practices.	"my observation is the tooling is quite limited. the challenge is that most of tools will have a lot of noise depends on the language you use and depends on certain features we use ,so a lot of time knowledge sharing is done with some kind of meetup platform where people just share Hey this is what we did, you should be using that. It will reduce amount of noise that you will have."
	Awareness of negative consequences of weak KM practices, Pride of current KM activities of Dev and Ops teams, Acknowledge of current knowledge sharing best practices	"the biggest challenge we have, it's like without a KM platform that glue things together, it is very difficult for different teams to figure out what is best practice, which kind of functions and which they should be using from this tool, which construction they shouldn't use… I believe it provides the kind of like maximum good experience"
	Awareness of negative consequences of weak KM practices,	"I think I don't think the problem is with tooling itself, I do think that's the process where we didn't have enough knowledge sharing and we didn't have enough training provided so that teams are struggling with writing Jenkins pipelines, we didn't really have a best practice in places."

7.8 Exemplary evidence of people-centered KM practices in DevOps teams

Table A7.8: exemplary evidence of people-centered KM practices in DevOps teams

Summary of quotes as first-order codes	Quotes from participants
Having intensive and daily interactions of Dev and Ops teams, Bringing together Dev and Ops teams for collaborative problem-solving	"We are sitting all close to each other and we can all share whenever we have an issue…Because we need something faster and we need something to react quick, face-to-face communication works well" "Face to face because sometimes describing an issue may take a long time and you take screenshots in Slack and all that things. It would be quick if we can ask in person what's happening here, what I'm seeing and if you want to see like console logs" "discussion and decision are of time where you do most knowledge sharing of some of the very interesting technical challenges. So what we do now is we have tech review sessions. What like basically the same the whole team, plus any stakeholders that might be the downstream consumers or upstream providers will come in to do the tech review to get them to make sure the knowledge is shared our property." "In design phase, we tend to have a conversation about it and where we will kind of talk about the topic and we'll propose an idea. Also, the team who might have some knowledge there might be able to say, oh, there's a risk that I can identify or other things like that." "The first line of communication that happens is that we have a Sync up meeting once a week, PM/UX/Dev (product management/ user experience/development), So that we're all know what's happening which also includes development and release engineers. So we have that first sort of meeting that is important." "we have meeting called after action review. we have meetings among DevOps members to talk about the technical problems. it is very important because after each meeting, we have some action item and new insight for future use" "we have an ADR process which is called Architecture Decision making Records. And you basically fill out a document about what the problem you're trying to solve, what the possible options were and

	why you chose the option chose and so we have an ADR that describes these are the two things, these are the differences. Now we have to select one and had a reason for it…it's good to have some idea of why we made a decision, because, you know, five years from now when it's a whole new team, different people work here and they're trying to figure out why we do this. That document is very helpful. We've only started doing that last year, this company, but definitely good process." "we are working based on the Scrum method. one part of scrum is sharing and socializing in a form of meetings (each day or every two days). In these meetings we have stand-ups and the members should say what did they do and what are they going to do. They can share their problems and challenges." "in DevOps team specifically that it's they are encouraged by every level of management to work with development teams and work with each other to ensure that the solution that's being provided as is appropriate and is scalable to what needs to be solved."
Encouraging self-driven learning practices, Exploring Dev and Ops Tacit knowledge voluntary practice	"A lot of training is also done is self-driven. learning a new technology and then somebody will investigate a new technology and then through meetings or documentation convey that information to the team." "So we need to have at least like bit of understanding of each of the services and find where could be the issue and what's happening and all that things. it gives you the chance to improve your skills gives you the chance to learn more and there is a lot of innovation. You see a real time problem you try to come up with the solution and it's always you like always being updated. There are always new things and they always looking to improve; you never stop improving." "There are some professional groups called Interest Group such as data science group, testing group. We have ten types of these groups. They meet each other every week or two weeks to discuss the technical issues. The interesting thing for most of these opportunities is that they are not compulsory, rather, it is up to the members to participate in these activities." "we have a mechanism called Guild and Chant program. So, the view is basically a group of people who have common interests on certain technology or certain topics; For example, there's a guild called

	Machine Learning, so the guild is cross-functional cross-teams. They would have regular meetings. They would have meet ups" "in RE team they have Lunch and Learn in which they watch a video prepared by team member and they show what they did." "We also have across the whole development team something they call Lightening Talks and what we do is making a presentation to the entire team about what you are working on. The company removed most of the barriers, you can talk without slides or specific structures… Those work very well because you can make a document. People don't necessarily read it but if they were in the presentation, they can get some information and for more details they can go to the written document." "We have an internal University where tons of meet ups, tech talks are happening." "In order to improve the knowledge capability, the company considers a training budget for each employee per month around 100$. I think this amount of money is a lot for training courses." "One thing that's very helpful in that process is engineers have a tendency to want to build the perfect solution to whatever problem. And perfect doesn't really go that well with time frames and getting things done in a reasonable time. So understanding how much you really need of the product to be completed for it to be functional help and solve the problem versus all of the things that you could potentially build into it, this is helpful to have feedback from the team." "There's a team internally called the Incident Management Team, actually, that makes sure that the remediation items are assigned to various teams, to the teams that need to work on them and are given a due date and follow up with those teams to ensure that the various remediation items are being done and taken care of and they keep track of that"
Implementing face-to-face mentoring sessions	"Mentoring, here the team creates a list of the abilities and needs of the employees. Based on this database, the employees can ask questions from each other. They arrange a short meeting to talk about that topic. This approach is new and very useful for me. I could say such mentoring is one of the strengthen of current KM practices." "There is training and mentoring that goes on there. For most part we have three very senior people so I would say there is definitely mentoring going on."

	"Whenever people have questions around those topics you will just go talk to the skilled and Chant members. So this is another way to facilitate cross team collaboration, which is not necessarily easily achievable by a traditional organizational structure." "Whenever we have new hires coming in, we provide a certain amount of training to those trainees. We also provide a continuous education, having different modules of different systems. …So my team will provide training material both in documentation platform as well as in-person training to make sure our engineers can get an necessary training to do their job."

7.9 Exemplary evidence of technology-centered KM practices in DevOps teams

Table A7.9: exemplary evidence of technology-centered KM practices in DevOps teams

Summary of quotes as first-order codes	Quotes from participants
Converting knowledge through communication/documentation technologies, Tracking and operating tasks in teams through collaborative technologies	"Slack is where people live and breathe these days. It's the fastest way to communicate. and we've tied all of our processes into it." "slack is incredible. It is so much better than any tool ever I've used before. So I've use all the one on one chat things and they have a problem because obviously collaborate super well. The idea, the mix of having the ability that channels and also individual chats and some files and all the things makes it does at one place for communication, that's very helpful for learning new things." "U-track and Jira, they provide an agile environment for tracking the problems. The interesting thing is that we can track the responsible person through the You Track for each assigned task. It shows the processes, the estimated time. We have to document the processes through this software even small tasks." "when we do release and there is an issue, we know which items are being released because of the You Track because it lets you know which of them are released. We have tags on You Track. We call it merge and release, we have a lot of bots that shows what are the merged items that not being released. And then when those merged items get released, we go, we change that, remove the merged tags so we have like that data when it goes to production." "We also use Slack a bit to support kind of the more engineering chat up workflows. And then we also have internal documentation and knowledge sharing thing that was high enough developed in order to kind of deploy documentation as a product to share with the rest of the company." "A lot of times we will follow up on our internal stack exchange and post things like add FAQ"

	"we have our own email exchange; we have our own instant chat rooms. I would say the email system, or the instant message system is probably the most important channel. We have confluence, we have a Stack overflow. We have Tweety as platform. People use JIRA for knowledge sharing and communication tools as well. So another very important documentation sharing kind of knowledge sharing is Google sheet, where we use a lot of like Google doc."
Automating the knowledge-related practices	"The increasing number of servers lead the company to automate the system. one person cannot manage such a huge number of servers." "It will automatically know that it's a web UI. It'll know that it's supposed to have certain things that it follows certain rules, and it will automatically be added into dynamic inventory so that we don't have to do any work to monitor a new server when we build it. And then similarly, if something were to go wrong, we don't have to do any. I don't have to notice something; it will alert us." "And all the alerts are Automate they go into slack and then slack notifies us everywhere all the time." "we have people who function as dev ops, who specialize in providing automation processes. Focus on getting the process smoother for development cycles." "once a quarter we will have a we call it like developers experience survey with both multi choices and open texts and try to get a sense of where other major points, basically which part of the whole workflow is least automated or people feel least confidence about." "we are moving more towards Prometheus and Grafana. And we do not insert images, and all happen very automatically now. So, we don't have to do a lot of manual labor there." "we also have a level of self-healing. So Megatron is what Rod and I wrote a couple years ago and what he does is he tests various things on the network and every minute of the time." "…in many cases remind you to move to a fully automated dev ops workflow which really automate pretty much everything. having the mindset to be can be changed in the

	sense that, you know, you don't necessarily have manual checks. So, automation works as lively having a testing in place.”
Converting knowledge through monitoring technologies	“I think in our team we need more specific monitoring tools rather than the other departments. We need to gather the real-time data and conduct the troubleshooting processes online.” “The real-time monitoring is needed to support automation to show alert and notification” “Real-time Monitoring. It is very important in our job. There are 5 dashboards now in the company so that every team member can see the alerts and problems on time and react immediately” “We are using some core tools and you can use them to monitor services for example using Prometheus. This is like to collect data and everything and using Graphene to create dashboards to know like the health of our system.” “So, when we're like when a bunch of teams are working on a single outage, we can create links to search and pass it around or a view of Metrix on Graphene and passed them around.” “we use the BI thing, so you can't retain instantaneous data for long periods of time because it's just enormous. So more aggregated and more longer-term data, we tend to roll up into BI kind of stuff where we use our own product to report on it.” “the thing that gathers telemetry, is called Prometheus, which is an open source tool. So you write your software to a report degree and Prometheus will go around and gather all the Telemetry that you deem. And then on top of that, there's another open source tool that we use called Graphene. And it is a way of doing a graphing and all this kind of stuff. So, you can actually see you can start to create alerts. You can observe trends and stuff like that.” “we use our own product to monitor and report the health of the system. one of our products is Power Metrics and we can track that over time so we can see for example how many people being heading the website. They do have another

	product that they use for tracking the users' interactions with the product that's call MIX Panel" "I have added many monitoring tools. For instance, in the past, we had many problems and we did not know how to prioritize them in our limited time. I made a dashboard that can give a query from our database and find the factors that cause problems in our system."

7.10 Exemplary evidence of alternative bridging KM practices in DevOps teams

Table A7.10: exemplary evidence of alternative bridging KM practices in DevOps teams

Summary of quotes as first-order codes	Quotes from participants
Alternative ways of doing KM practices is a mindset for DevOps	"what if slack is down? we're using Google Hangouts, so we are having a production outage, we also made sure that all of our self-healing bots, which usually we control through slack, can be done and we can either talk to them via rest API or we can send command line notes to them." "we use Confluence for documentation and things like that, it's just as a way to bridge the gap when we aren't able to fully automate things." "It's not worth integrating into the alternative options we just have to be able to do it without slack. So, it's not as convenient if Slack isn't there but we don't have to have slack running,"
Using KM features embedded in Dev and Ops technologies	"documentation and the communication within tool, it is very effective because it's part of a workflow we're trying to do as much as possible" "Git Hub, a code development tool, has something called a Pull Request (PR). So, I write some code …you tag a couple of people who are knowledgeable about this area and they will look over your code … it's nice to have a couple extra sets of eyes on things. People learn a lot from that" "When you create code, it gives you a big box say here is like describe the changes describe what you did within the GitHub environment." "Each of us is working on one specific code. If we do not have GitHub, then how can I share my file with team. It would be hard to email or share such file on network. We put our files on GitHub repository."
Building integrated and continues KM platform	"We have official documents platform, which is in-house technology, which is open source within our company. So basically, this platform is, you just write some GitHub pages and then we'll look at other XML to render that documentation page. So, within the platform, whenever there is an issue, typically you would have a linkage to official documentation"

	"most of the documentation, discoverability or knowledge information discoverability it is within the platform itself… all the tools are integrated by a platform our organization provides." "we have structured knowledge management platform… so the off the shelf platform I mentioned, I believe it provides the kind of like maximum good experience So that good balance between the details you need to know about the specific process versus all the details of the tool itself." "we have internal documentation engine… That's where I looked for documentation. And I think that's where a lot of the company is starting to lean. Unfortunately, that's not like an open source tool, say people should use this one. I think the lesson learned is that treating documentation like a result, like a releasable thing, I think is the direction we need to head."

7.11 DevOps KM practices: characteristics and examples from data

Table A7.11: DevOps KM practices: characteristics and examples from data

Characteristics of KM Practices	Examples of DevOps KM Practices
People-centered KM practcies	• Face-to-face meetings and stand-ups • In-person pair programming • Collaboration between mentors and apprentices • Production post-mortems • Daily scrum meeting • Sync Up meetings • Lunch and Learn meetings • Lighting talks • Interest Group program • Less formal communications • exploring Dev and Ops tacit knowledge through voluntary practices • self-driven learning practices • bringing together Dev and Ops teams for collaborative problem-solving
Technology-Centered	• communication/documentation technologies such as • collaborative technologies • project management tools • Automating the knowledge-related practices • Architecture Decision making Records (ADR) online forms • Instant Message System • Microsoft teams • Stack Exchange • Google sheets • Share Point intranet portal • You Track • Jira • Ansible • SMS • Slack alerts bots • monitoring technologies • Learning by doing via technologies such as online courses

	• New technology with case study • Developers self-service environment and deployments. • Using shared log files and joint platform • wiki-based communication networks • Online Knowledge Repository • Synthesis feedbacks in large-scale database
Alternative bridging KM practices	• KM features embedded in Dev and Ops technologies • Git Hub PR • integrated and continual KM practices and tools • building in-house KM platform