Exploring the Challenges and Barriers of Knowledge Retention in Information Systems Development Teams: The Case of Pēke

Abstract

Knowledge Retention (KR) is vital for information systems development (ISD) as information technology (IT) professionals rely on accumulated technical and organizational knowledge to develop and maintain information systems. To help organizations better understand the barriers and challenges to KR in the ISD context, we explore the attrition of KR practices arising from staff churn and the aftermath of a major earthquake on an ISD unit from a financial organization in New Zealand. In this preliminary study, we develop a causal model of KR in the ISD context, which articulates the barriers, challenges, and consequences of ineffective KR for at the routine and exiting stage. Our model identifies four barriers and challenges—coordination complexity, resources for knowledge retention, attention to knowledge retention, and process for hiring and handover—which can affect the loss of ISD knowledge when routine and exiting KR fall into disarray. Moreover, we offer implications for practitioners regarding KR in the ISD context.

1. Introduction

The IT industry has been at war for talent over the decades. The median employee tenure in a large multinational IT corporation is 3.45 years, ranging from 1.1 years at Facebook to 7.1 years at IBM [1]. Faced by increasing pressures to become and remain digitally agile, organizations can incur a steep cost in the form of competitive advantage erosion and inferior firm performance if critical knowledge is not transferred and retained [2]. Knowledge retention (KR) has become a strategic risk for many organizations nowadays, and this risk is rendered particularly acute in organizations facing a greying IT workforce [3], such as small firms [4], as well as organizations that rely on external labor and consultancy markets to fill talent shortages [5]. To counteract such risk, organizations (1) adopt numerous human resources practices to improve talent retention (e.g., work environment design, career development, employment incentives) [6] and (2) implement knowledge management (KM) practices (e.g., knowledge ownership practice, knowledge exchange policy, debriefing after important events and projects) to orchestrate knowledge within their pool of talent [7].

Yet, although the factors that drive successful talent and KM practices have been well researched over the last two decades [8], surprisingly sparse attention has been paid to the barriers and challenges organizations are facing in setting up successful practices for KR following IT personnel or contractor turnover, with exception to some work [9]. It is still a common organizational experience for outgoing IT experts to be submitted to rushed exit interviews and for newcomers to be bewildered by unstructured handovers [9], resulting in inefficient KR. To retain such critical knowledge in the long-term, organizations need to prioritize the types of knowledge to be retained, evaluate the risks of knowledge loss (KL), and overcome the obstacles for implementing KR practices [10]. KR is particularly essential for information systems development (ISD) because information technology (IT) professionals rely on various types of ISD knowledge (e.g., how different types of hardware and software are configured; the organizational context in which the system is developed and used) to develop and maintain IT [11]. Losing ISD knowledge prevents IT professionals to deliver value to an organization [12] and cripples IT-dependent organizational agility [13].

The purpose of this preliminary study is to identify key barriers and challenges to KR in ISD project teams. With a case study of an IT unit in a financial organization (Pēke) in New Zealand, we develop a conceptual model that identifies (1) factors hindering the performance of KR in organizations and (2) KL associated with ineffective KR in ISD project teams and their members. Our study contributes to the literature on KR in the ISD context by proposing a novel practice-based conceptualization of the barriers and challenges that contribute to KL when routine and existing KR is dysfunctional. In the discussion, we explore the implications for the development of a practice-based perspective on KR in ISD project teams, which informs organizations on how to prevent slipping into a cycle of KL.

2. Conceptual Background

KR is concerned with preserving and maintaining the knowledge embedded in individuals and their relationships with others to cope with challenges arising from the exit of employees [10, 14]. It can be considered as a special form of knowledge transfer, which occurs "*when knowledge has been transferred from a knowledge owner to the organization and can be reused by a knowledge seeker*" [15]. It serves the purpose of transforming knowledge between individuals and their organization at two stages: (1) *routine KR activities* (e.g., transfer, capture and storage of knowledge of existing experts within the organization) and (2) *exiting KR activities* (e.g., transfer and storage of knowledge held by those departing from the organization) [16]. While various KM strategies, including IT-oriented and people-oriented, have been proposed to enable KR, KL becomes inevitable either because knowledge is not retained properly or too costly to reuse [10, 15, 17]).

KL can be attributed to many reasons. Broadly speaking, it can be categorized into the intentional or unintentional disappearance of knowledge, which has been accumulated from learning and from individual and collective actions [18]. In our study, we address the unintentional KL in the context of ISD. Prior literature has identified different drivers of KL, including ineffective organizational routines and memory [19] and employee turnover [10], which have negative impact on the organizational performance. Furthermore, different explanations were provided supporting the view that KL might be beneficial when an organization unlearns wrong routines, which are obstacles to effective acquisition and absorption of valuable new knowledge, or detrimental when ineffective memory systems [19].

In the context of software development, KR provides the necessary conditions to incorporate prior knowledge and experience into the innovation development process [20]. ISD knowledge to be considered in KR includes technical knowledge related to the IS applications and their underlying technologies, as well as organizational knowledge, including processes and structures [11]. Technical and organizational knowledge, such as architectures, databases, and business rules, is explicit and easily captured and retained [21]. Other knowledge is tacit and experiential due to the complex, abstract, and context-dependent nature of ISD work. For instance, clients' needs are implicit and volatile [22] and thus require a substantial amount of interaction to be understood [23]. The content of tacit knowledge can also be technical in nature and reside in both individuals and teams [24]. For example, local coding conventions and design practices often reside among experienced programmers and are difficult to be transferred to newcomers [25]. As well in the team environment, which involves team of teams, multi-sourcing, and distributed working environment, team members should understand not only who possesses what specialized knowledge [26] but also how to coordinate fluidly [27].

Previous studies on factors that enable and inhibit KM [28] guide the investigation in the KR context. Individuals are either intrinsically (e.g., personal development, learning, and recognition) [29] or extrinsically (e.g., obtaining retention bonuses) motivated to contribute to KR [30]. Individuals' positive or negative attitudes toward KR is also important. For example, when sharing knowledge weakens one's power and jeopardizes his or her job security, individuals tend to have negative attitudes and are less likely to contribute to KR [31]. Leadership is crucial to KR. For example, leaders should set a strategic priority for KR, engage in KR initiatives, and build a culture valuing knowledge sharing [32], which in turn raises the awareness of what knowledge should be retained and why it is important.

3. Research Methodology

Because KR in ISD projects represents a setting where the intertwined relationship between ISD professionals and the context should be considered together, we conducted a case study, with the use of the inductive and deductive approach to propose a framework [33]. All our data was collected from the IT department, in particular, the development unit, of a financial organization in New Zealand (Pēke), which was purposively selected as a revelatory case because of its conceptual potential to highlight the difficulties involved in setting up KR practices in ISD project teams.

3.1. Case Background

Pēke offers customers a range of personal, business, and international financial services. IT has continuously played an important role in Pēke, enabling to provide convenient facilities through a diverse range of digital channels or platforms. Pēke's IT department consists of numerous project teams of varying sizes; thus, integration and coordination are necessary to deliver IT value. Therefore, it is essential for the project teams to work together to implement effective IT solutions, which involves (1) fulfilling legislation and compliance requirements,

(2) supporting existing systems or applications, (3) improving applications with advancing technology, and (4) implementing new business functionalities to serve customers and employees. At the time of our study, there had been a significant churn of resources at Pēke's IT department. For many years prior to the study, this IT department had been composed of permanent and contract developer positions. A significant number of developers who had been recruited on a contract basis ended up working for 3 or 5 years, leading to an increasing dependence on their expertise. Not long before our study, Pēke's senior management terminated the employment of a significant number of these contract positions to reduce costs. This sudden downsizing halved the size of the IT project teams in a very short time. Although some contract positions were replaced with permanent positions over the course of the year, many positions were still vacant as it has been difficult to recruit developers with the required skillset.

Following the downsizing, most ISD developers were new to Pēke with less than a year of experience in their present roles. As long-term contract developers left, they took the knowledge and expertise they had built up over time with them. The loss of experienced developers has meant that the teams also lost a lot of critical information around developing, supporting, and delivering IS solutions. Furthermore, the situation was compounded after an earthquake of magnitude 7.8 Mw shook much of New Zealand in late 2016. As a result, interaction and collaboration in project teams and across the IT department became increasingly difficult. Prior to the earthquake, all of the project teams were co-located in one city; however, after the earthquake, the teams were rapidly dispersed across multiple locations in two different regions. The dispersed operations compounded the impact of KL and created new challenges with team processes, particularly regarding communication and alliance between developers in different locations. Following the unit's dispersal, the developers principally relied on email or instant messaging for inter-unit collaboration and coordination.

3.2. Data Collection and Analysis

We gathered evidence from multiple sources—interviews, company documents, and observation notes—that we used to identify factors hindering KR and the impacts of KR. We conducted six semi-structured interviews in August and September 2017, lasting between 40 minutes to 1 hour, with two inexperienced developers (developers A and B), four experienced developers (developer C, D, E, and F) (see Table 1). We have selected to interview both experienced and inexperienced project members to develop an in-depth understanding of the KR challenges and barriers associated with departing and new developers and project in general. The total interview duration time was approximately 5 hours, approximately 40,000 words over 65 pages. After each interview, we also wrote observation notes to capture our impressions while they were fresh and had several informal conversations with the respondents that resulted in 5 pages (approximately 3,000 words) of notes. We used company documents to understand official organizational processes and perspectives. Altogether these sources of evidence provided us with an in-depth understanding of Pēke's ISD teams and their challenges to KR.

Table 1. Interviewees' Demographics

Interviews	**Working experience**	**Job Role and ISD Team Information**
Developer A (Female)	4 months	A full-stack developer; Member of a small development team that is composed of relatively new members (i.e., less than 2 years' services).
Developer B (Male)	8 months	Technical lead of the team; Member of a small development team that is composed of relatively new members (i.e., less than 2 years' services).
Developer C (Male)	5 years	Technical lead of the team; Member of a medium-sized development team that is composed of relatively experienced members (i.e., more than 5 years' services).
Developer D (Male)	9.5 years	A full-stack developer; Works alone, without membership into any team. He is the only developer who supports several IT applications.
Developer E (Male)	5 years	Leads technical delivery; Interacts with several development teams of varying sizes that are composed of new and experienced members.

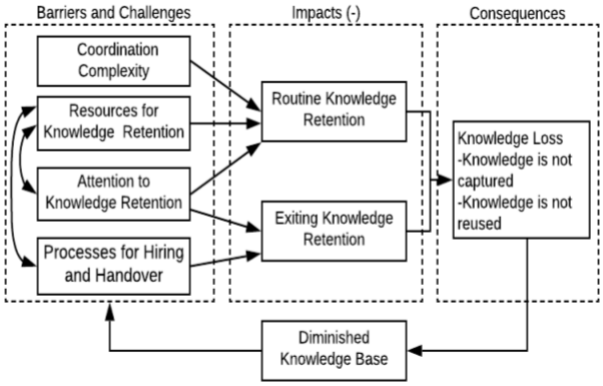
Developer F (Female)	10.5 years	Leads system analysis and design; Interacts with several development teams of varying sizes that are composed of new and experienced members.

We transcribed all the information that we gathered including interviews and observation notes. We then adopted the template analysis technique [34] for a thematic analysis using NVivo 10. Our analysis was sensitized to an extent by knowledge of the knowledge-based theory of the firm [35], but we allowed codes to naturally emerge from the data (i.e., grounded theory coding). We constantly modified codes throughout the analysis based on their usefulness and suitability, which resulted in the modification of several themes. The final coding template consisted of 4 main themes and 18 sub-themes related to the individual and organizational barriers and challenges, routine and exiting KR activities, and KL. We presented key findings in the following sections.

4. Findings

Our analysis led us to the development of a practice-based conceptual model that depicts the factors that conspired to prevent the performance of KR practices, and the associated consequences of KL in the ISD team context (see Figure 1). We identified two distinct conceptualizations of KR: (a) routine KR and (b) exiting KR. *Routine KR* refers to those activities that involved information sharing within and across ISD teams, as well as the division of labor among ISD teams on a regular basis. *Exiting KR* refers to activities that involved individual IDS team members' handover practices, hiring practices, as well as knowledge documentation and archiving practices. At Pēke, all of those practices had become rare/endangered and were even seen as undesirable (and time-consuming) by those employees who were busy trying to solve the problems arising from KL, and who had a problem-solving orientation rather than a knowledge sharing orientation. Another key finding from our analysis was that barriers and challenges can lead to knowledge not being recorded at both stages, and also prevents knowledge from being reused. The conceptual model summarizes the casual relationship between the identified barriers and challenges, the diminishing of KR activities, and KL. These findings are outlined in detail in the following sections. We provide exemplary evidence for barriers, challenges, and subsequent consequences in Tables A1 and A2 in Appendix.

Figure 1. A conceptual model for KR in ISD teams



4.1. Barriers and Challenges of Routine KR Practices

4.1.1. Coordination complexity

Coordination complexity in the form of work dispersion prevents knowledge retention. The multiple dispersed work sites, particularly after a major earthquake, contribute to reduced interactions and communication between developers, making it harder to work collaboratively and share knowledge. Furthermore, the project teams, including development and operations teams, that need to work together for system integrations when implementing new or updating IT functionalities are currently not co-located, which causes developers to work in silos. Developers who are in the same unit work independently rather than collaboratively. The complexity of coordination further increases as a result of fragmented task allocation. Developers in the same unit work on different areas of a single application. Hence, each developer has a focus area that they develop and support rather than the entire application. Since the systems at Pēke are tightly coupled, developers in a team must develop technical and organizational knowledge to complete tasks via interpersonal communication. Such communication often requires experts external to the team. Ad-hoc, sporadic communication with external team members, along with dispersion, compounds routine KR. The recent downsizing of teams exacerbates the KR issue. In situations where there is only one developer who assumes both development and maintenance roles for several IT, ISD knowledge is barely retained. Knowledge about those critical systems is only held by an individual. Because such developers do not have the privilege of working with others as part of a team, they don't have anyone to share or transfer their knowledge to.

4.1.2. Insufficient attention to KR

Insufficient attention to knowledge retention from management has been a major issue. Knowledge retention is not considered as part of developers' role and responsibility as there is very little prioritization for KR and encouragement for sharing from the management. KR is not the strategic priority of Pēke. In fact, people who used to facilitate KM were disbanded. Without the push from management for KM practices, developers are not motivated to transfer their accumulated knowledge to others. Besides management's claims that KM is important for teams' information flows, developers felt that hardly any constructive actions had been taken to encourage KR within the teams. Developer C commented: *"[m]anagement doesn't support KM. Also, they don't know where that knowledge sharing will lead to productivity somehow because those are intangible benefits and not many people can actually measure those kinds of benefits and may not be recognized so easily."* Over time, developers possess DIY attitudes, orienting towards solving problems on their own.

A lack of incentives and encouragement reinforces knowledge hoarding behaviors. Developers try to remain to be the key person or the expert of the applications they maintain. Senior developers in particular that solely support legacy systems have hoarding tendencies as they are reluctant to share their knowledge to keep their positions secure in the organization. Being the only person that knows how the system works means that they have control and dictate the development, deployment and support of it. Similarly, such developers have difficulties working in a team environment.

4.1.3. Insufficient resources to KR

Workload pressure and lack of time hinder knowledge retention in project teams. Respondents point out that, due to resources churn in the IT department, existing developers have increased workload, trying to fulfil the responsibilities of developers that have left. Additionally, with the volume of project work prioritized by management, developers do not have sufficient time for documenting or sharing knowledge with their peers.

4.2. Barriers and Challenges of Exiting KR Practices

4.2.1. Rigid processes for hiring and handover

The hiring process for permanent positions takes excessive time due to the required approvals from various organizational levels, which adds to the rushed handover process. New developers usually join only a few days before departing developers leave, which does not give sufficient time for adequate knowledge transfer. Developer D complained that: *"[o]ne of the worse handovers I had that was given to me was two hours before the guy left the company."* Sometimes, a position is filled after the developer exits Pēke, which in turn significantly decreases the quality of knowledge transfer.

Related to insufficient resources for KR, new developers note that when employees depart, their workload during their notice period increases, as they are expected to quickly complete all the pending tasks on their plate before leaving. They still do their daily work right through to the point when they exit. With the increased workload, the departing developers find it difficult to share knowledge before leaving the company. Moreover, developers don't invest adequate time or effort during their notice period as knowledge transfer is not prioritized over project work.

New and existing developers find the handover process to be rushed and dense as many new developers usually have only one or fewer hours of formal handover in the induction. New developers are overwhelmed with dense and unstructured knowledge. Developer E found that even when newly hired developers are experienced, handover can be challenging because of unique, complex development environments. He gave an analogy for handover between developers: *"[i]t's not just like speaking French when you speak to a French person. It's actually, you know, you'll be speaking French to a, to a French neuroscientist."* It is also noted that the handover process provides only very basic ISD knowledge that is not very useful for solving complex issues spanning multiple systems and involving multiple business users. Developer B indicated that: *"[f]rom the initial training - not really, there was just a lot of dense training and I had to dig into the code to figure it out myself... we didn't have any visual documentation on where things are or for incidents is a good example was there was no documentation on how we solve them so then we had to rework it out when they came in."*

The handover process is not structured, and a lot of crucial information is missed depending on the departing developer's time and personal motivation. Developer E said that: *"[t]he information exists, but it's not structured, so lack of structured information so what that means is, there could be 1,000 documents detailing everything you need to know about the system, but it's got no categorization or structure and not in one place."*

As departing developers are usually under workload and time pressure, they are too busy to help or adequately transfer knowledge to new developers. Such informal knowledge transfers provide only basic and high-level information about IT.

4.2.2. Insufficient attention to KR

New developers acknowledge that there is a lack of any formal KR policies or process for capturing, sharing, and transferring of accumulated work knowledge when they join the team. IT managers never prioritize KR as there is always a higher priority for developers to focus on. This is a significant issue when current developers are departing, and new developers are inducted. KR is usually not planned effectively in which departing developers have a four-week notice period to handover their responsibilities to the new developers. New developers are not formally trained to capture and share their knowledge, neither utilize any mechanisms or technology.

New developers acknowledge that there is a lack of useful, relevant, and up-to-date documentation created by departing developers for them to refer to. Although numerous documents are available on the intranet share-point sites, most of them are not considered to be current or relevant any more. There is a lot of documented information that is out of date as the applications have changed significantly since then. Documentation regarding business and functional requirements of the project and architecture design are documented before starting development for approvals and sign-offs. However, due to lack of archiving process and prioritization for documentation after commencing development, these documents are not updated regularly as the project evolves. Developer A felt that: *"it's like they have just created some documents for the sake of having documents, but not in terms of full knowledge."*

New developers reported that there was a lack of documentation created by departing developers particularly around the high-level architecture of IT and their integrating components. Also, information is not documented regularly for support or maintenance of the applications. Thus, documentation is limited and only produced when it is required with insufficient information. Often new developers are not aware of documentation created by departing developers, which exists regarding their application or project as they were not told about it. Additionally, there is a vast number of documents on share-point site that are not categorized or structured properly for easy access. The documentation is stored randomly with different teams and developers following their own practices for documentation.

4.3. Knowledge Loss

4.3.1. Knowledge is not captured

KL is a major concern among all the interviewees. As experienced developers leave the organization, the knowledge they had gained over the years regarding development, integration, deployment, and maintenance of critical IS components is being lost. Valuable understanding of the systems, applications, and their business functions is not being retained within the organization when developers depart. Thus, critical technical and organizational knowledge is lost when developers leave as they are not able to provide sufficient guidance to existing or new developers to acquire the knowledge needed to take over their responsibilities. Developer A felt frustrated with a lack of knowledge: *"I did not find any document, maybe the documents are not there or maybe the proper sessions for providing the information like what is the full architecture of the project, how where the project stands among the other projects in [Pēke], sometimes apart from the development understanding the business logic is important, we also need some other stuff like release management, deployment and all. So, these things are still missing."*

New developers felt that the knowledge transferred during their handover process was insufficient. They lacked knowledge about the full system architecture and its wider context in order to understand key integrations between systems that are necessary. Understanding these aspects are essential when implementing IT as developers need to recognize the impacts on business processes and dependent systems when updating a piece of code. Moreover, without adequate documentation and poor handover process, new developers also lacked understanding about IS build, versioning, and release processes necessary for deploying IT to different environments. Understanding the business context is crucial for developers to realize the business

processes, functionalities, and requirements it satisfies. Developer F emphasized the importance of multiple aspects of knowledge required for ISD: *"[w]hen you have an issue in production, when you look at the code, it will give you part of the story, but the rest needs to come from our channels or our customers. And it's not really documented anywhere."*

Moreover, in the area of legacy systems where talent is hard to acquire, leading to key personnel risk. In one-developer teams, the developer is forced to provide support when they take long holidays as there are no other developers with even the slightest knowledge about the systems to support if required. Developer D indicated that he has to be on call all the time because there is no backup. He has no intention and time to follow routine knowledge retention activities. With the rapidly changing landscape of programming and technology, it will be very difficult to recruit new developers with relevant skillsets to support legacy systems as much of the developers available in the current market would not have experience with obsolete technology.

4.3.2. Knowledge is not reused

We observe that knowledge is not reused due to the following reasons:

(1) Lack of confidence in reusing captured knowledge

Developers lack confidence in their new role because they do not have a complete understanding of IT they are required to support. They are often worried and uncomfortable making changes to the codebase and deploying those changes as they are uncertain about their impacts on the business processes and other systems. They are hesitant to reuse knowledge captured in the system. Developer C commented that: *"if you lost senior developer then you start to worry, and other employees will start to feel uncomfortable doing changes in the area and lack of confidence and will take a long time to build confidence and to gain the knowledge."* New developers are thrown into the deep end where they are expected to pick up responsibilities without adequate documentation and training from the developer they are replacing. They point out that they struggle with locating relevant information and individuals within the organization that can help them solve an issue faster. Significant time is spent on searching for knowledge which in turn prevents new developers from acquiring the necessary skills to efficiently and effectively work on their given tasks.

Developers take a long time to learn and build up the appropriate level of knowledge to be confident and effective in their new development role. They usually need to dig through the codebase or search for a key person to figure out aspects of IT, which is often time consuming and inefficient. Developer B found out that: *"[a] lot of key information was missing that made something that could be simple with a bit of training, so it means that its 10 hours of work to work it out as opposed to getting it solved in 15 minutes."* Developer F further added that: *"[w]e have two developers on our team. They're actually amazing senior developers. They are, I, I call them partial BAs [business analysts] and partial devs. And they will go out to the business, and they take, they put a lot of time and effort into understanding the process. They will not do anything until they understand."*

Similarly, because developers are not fully aware of the implemented or existing features, they end up re-inventing the wheel instead of reusing functionalities, resulting in re-implementation. This is because re-implementation can sometimes be easier and quicker than searching complex codebase or finding someone that can provide the information. As Quality and security are critical for Pēke, starting from the scratch is consider as a low risk pathway.

(2) Difficulty in locating knowledge

Even when knowledge is captured in the organization, tracking down key developers with appropriate knowledge about a specific feature is one of the main difficulties for newcomers. When developers require additional information, to successfully implement and deploy IS, they often have challenges attaining relevant and correct information on their own as most of the documentation are out of date. Due to knowledge silos, information about a feature is usually trapped in one individual developer's mind, making it difficult to attain that information. New developers often feel lost when trying to find the right person with the right information and take a very long time approaching numerous individuals in this search process. Developer F indicated that: *"you always need to go to key person to understand how do you do this, what are the branches, which environments to connect to etc. it's all about knowing people, networking basically for each component."*

Social connections and personal relationships are a fundamental aspect of the IT department culture where developers need to establish networks and have relevant contacts within the IT department to be effective. Knowledge silos have led to a 'shoulder-

tapping' culture as it is often crucial to know the right people for knowledge and information to successfully integrate systems and deliver IT solutions. Without having the right connections to "shoulder-tap" when needed, it becomes difficult for new developers to work well without having any other access to reuse the relevant information.

5. Discussion

Our findings illustrate an organization trapped in a vicious cycle of KR. From a strategic choice perspective, it starts when an organization does not recognize the value of KR. Once critical knowledge slips away, "organizational amnesia" puts pressure on newcomers who constantly search for missing knowledge and even need to reinvent the wheel, jeopardizing organizational IT capabilities. Particularly, newcomers suffer from missing technical knowledge in terms of IT architecture, which is inextricably intertwined with organizational knowledge. The two aspects of ISD knowledge are difficult and costly to rebuild once they are lost in the process. Our model identifies factors, including the organizational structure dimension (i.e., coordination complexity) and the managerial dimension (i.e., insufficient attention from management and insufficient resources), which can affect the loss of ISD knowledge when routine and exiting KR fall into disarray. Considering high fluidity of organizations nowadays, we believe that these inhibiting factors are prevalent. However, to go beyond the single case in this study, future longitudinal and comparative research can validate our observations. They can also extend attention to the contingencies, such as environment volatility, scarcity and munificence, organizational slack, and organizational turnover.

We suggest that organizations carefully devise a KR strategy and build KR into the routine and exiting stage. KR practices can be implemented through personalized approaches, such as mentoring, storytelling, and oral histories, [14], along with KM systems, such as the electronic community of practices and knowledge repositories [7]. Recent advancement of intelligent software agents, such as Documentation Bots and DevOps Bots, shows potential to capture critical knowledge at the routine stage [36]. Routine KR practices reduce the pressure when limited time available for exiting KR. For instance, making routine KR aligning with performance reviews or KPIs will help reinforce organizational values. Building routine KR culture should not only rely on extrinsic rewards, but also collaborative culture. For instance, knowledge silos can be broken down by regular communication and job rotation [37]. Such routines can ease the process of knowledge identification and boost the confidence of knowledge reuse.

Exiting KR is more challenging, as departing experts are constrained by time and lack of motivation to transfer knowledge to organizational memory. Structured exit interviews with a focus. Moreover, highly prioritized knowledge areas can minimize KL. What complicates the exiting process further is that IT managers and ISD project team members may have a hard time identifying what critical knowledge is worth retaining. Our study didn't assess what knowledge the IS senior managers or business stakeholders of Pēke considered to be critical to retain, and what was the position of the ISD in the financial institution's portfolio of IS capabilities. Thus, multi-level research designs seem to have promise for future research wishing to extend and refine our findings.

Yet, our findings also point to a somewhat more dismal implication for some organizations. ISD project teams in organizations that are resource-starved, either due to environment scarcity or to managerial frugality, may find escaping the path-dependent trajectory of ineffective KR practices difficult because these practices become self-reinforcing over time. It is likely that Pēke's misfortunes have their origins in the prior strategic choice of sourcing ISD talent from external labor markets, which led to the erosion of an internal ISD capability, putting the organization on an evolutionary trajectory. Such strategic choice, if not complemented with appropriate practices to retain knowledge, can thus contribute to perpetual patterns of firefighting [38].

6. Limitations and Opportunities

As this is a preliminary study on the KR barriers and challenges in the ISD context, our findings have some limitations that provide opportunities for future research. First, while particularly revelatory, the strong contextualization is both a strength and limitation of our study. The generalizability of our findings to other settings could be limited as factors such as industry composition, infrastructure, or culture might play a role. We thus encourage researchers to test the generalizability of our model not only in other industries but also in other geographical regions and cultures. Second, it would be useful to test our proposed model with larger samples to establish the validity of our model and further refine it. Large-scale surveys or analysis of secondary data are two potential ways to do so. This

also opens the opportunity to define and refine measures that are specific to our model, and that will support our key themes.

7. Conclusion

In our study, we theorized about the key barriers and challenges for KR in ISD context using a case study of an IT unit in a financial organization, Pēke, in New Zealand. Based on inducive and deductive thematic analysis of rich interview data, we develop a practice-based conceptual model that identifies (1) factors hindering the performance of KR in organizations and (2) KL consequences associated with ineffective KR in ISD project teams and their members. Our study contributes to the literature on KR in the ISD context by introducing a novel practice-based conceptualization of KR comprising two stages.

8. References

[1] PayScale. "PayScale Data Packag: Connecting the dot-comes." https://www.payscale.com/data-packages/top-tech-companies-compared (accessed 28th of Feb., 2018).

[2] A. Daghfous, O. Belkhodja, and L. C. Angell, "Understanding and managing knowledge loss," *Journal of Knowledge Management,* vol. 17, no. 5, pp. 639-660, 2013.

[3] R. V. D. Gonzalez, "Knowledge retention in the service industry," *International Journal of Knowledge Management,* vol. 12, no. 1, pp. 45-59, 2016.

[4] S. Durst, G. Bruns, and I. R. Edvardsson, "Knowledge Retention in Smaller Firms," in *Contemporary Knowledge and Systems Science*, W. B. Lee and F. Sabetzadeh Eds. Hershey, Pennsylvania: IGI Global, 2018, pp. 100-119.

[5] P. Cappelli, "Talent management for the twenty-first century," *Harvard Business Review,* vol. 86, no. 3, pp. 74-81, 2008.

[6] R. Agarwal, C. V. Brown, T. W. Ferratt, and J. E. Moore, "Five mindsets for retaining IT staff," *MIS Quarterly Executive,* vol. 5, no. 3, pp. 137-149, 2006.

[7] J. Kotlarsky, H. Scarbrough, and I. Oshri, "Coordinating expertise across knowledge boundaries in offshore-outsourcing projects: The role of codification," *MIS Quarterly,* vol. 38, no. 2, pp. 607-627, 2014.

[8] K. Dalkir and M. Beaulieu, *Knowledge management in theory and practice*. MIT press, 2017.

[9] C. R. Coombs, "Improving retention strategies for IT professionals working in the public sector," *Information & Management,* vol. 46, no. 4, pp. 233-240, 2009.

[10] C. E. Martins and H. W. Meyer, "Organizational and behavioral factors that influence knowledge retention," *Journal of Knowledge Management,* vol. 16, no. 1, pp. 77-96, 2012.

[11] J. Iivari, R. Hirschheim, and H. K. Klein, "Towards a distinctive body of knowledge for Information Systems experts: Coding ISD process knowledge in two IS journals," *Information Systems Journal,* vol. 14, no. 4, pp. 313-342, 2004.

[12] T.-C. Lin, C. L.-h. Chang, and W.-C. Tsai, "The influences of knowledge loss and knowledge retention mechanisms on the absorptive capacity and performance of a MIS department," *Management Decision,* vol. 54, no. 7, pp. 1757-1787, 2016.

[13] L. Fink and S. Neumann, "Gaining agility through IT personnel capabilities: The mediating role of IT infrastructure capabilities," *Journal of the Association for Information Systems,* vol. 8, no. 8, pp. 440-462, 2007.

[14] J. Liebowitz, *Knowledge retention: Strategies and solutions*. Boca Raton, Florida: CRC Press, 2008.

[15] N. Levallet and Y. E. Chan, "Organizational knowledge retention and knowledge loss," *Journal of Knowledge Management,* 2019.

[16] M. Levy, "Knowledge retention: Minimizing organizational business loss," *Journal of Knowledge Management,* vol. 15, no. 4, pp. 582-600, 2011.

[17] P. R. Massingham, "Measuring the impact of knowledge loss: a longitudinal study," *Journal of Knowledge Management,* 2018.

[18] B. E. Perrott, "A strategic risk approach to knowledge management," *Business Horizons,* vol. 50, no. 6, pp. 523-533, 2007.

[19] P. M. d. Holan and N. Phillips, "Remembrance of things past? The dynamics of organizational forgetting," *Management Science,* vol. 50, no. 11, pp. 1603-1613, 2004.

[20] P. H. de Souza Bermejo, A. O. Tonelli, R. D. Galliers, T. Oliveira, and A. L. Zambalde, "Conceptualizing organizational innovation: The case of the Brazilian software industry," *Information & Management,* vol. 53, no. 4, pp. 493-503, 2016.

[21] D. Damian, "Stakeholders in global requirements engineering: Lessons learned from practice," *IEEE Software,* vol. 24, no. 2, pp. 21-27, 2007.

[22] A. Kankanhalli, F. Tanudidjaja, J. Sutanto, and B. C. Tan, "The role of IT in successful knowledge management initiatives," *Communications of the ACM,* vol. 46, no. 9, pp. 69-73, 2003.

[23] S. Chakraborty, S. Sarker, and S. Sarker, "An exploration into the process of requirements elicitation: A grounded approach," *Journal of the Association for Information Systems,* vol. 11, no. 4, pp. 212-249, 2010.

[24] S. Ryan and R. V. O'connor, "Development of a team measure for tacit knowledge in software development

teams," *Journal of Systems and Software,* vol. 82, no. 2, pp. 229-240, 2009.

[25] I. Rus and M. Lindvall, "Knowledge management in software engineering," *IEEE Software,* vol. 19, no. 3, pp. 26-38, 2002.

[26] J. He, B. S. Butler, and W. R. King, "Team cognition: Development and evolution in software project teams," *Journal of Management Information Systems,* vol. 24, no. 2, pp. 261-292, 2007.

[27] D. E. Strode, S. L. Huff, B. Hope, and S. Link, "Coordination in co-located agile software development projects," *Journal of Systems and Software,* vol. 85, no. 6, pp. 1222-1238, 2012.

[28] A. Riege, "Three-dozen knowledge-sharing barriers managers must consider," *Journal of Knowledge Management,* vol. 9, no. 3, pp. 18-35, 2005.

[29] A. Kankanhalli, B. C. Tan, and K.-K. Wei, "Contributing knowledge to electronic knowledge repositories: an empirical investigation," *MIS Quarterly,* vol. 29, no. 1, pp. 113-143, 2005.

[30] J. Bairi, B. Murali Manohar, and G. K. Kundu, "Knowledge retention in the IT service industry," *Journal of Systems and Information Technology,* vol. 13, no. 1, pp. 43-65, 2011.

[31] M. M. Wasko and S. Faraj, "Why should I share? Examining social capital and knowledge contribution in electronic networks of practice," *MIS Quarterly,* vol. 29, no. 1, pp. 35-57, 2005.

[32] P. Gelard, Z. Boroumand, and A. Mohammadi, "Relationship between transformational leadership and knowledge management," *International Journal of Information Science & Management,* vol. 12, no. 2, pp. 67-82, 2014.

[33] K. M. Eisenhardt and M. E. Graebner, "Theory building from cases: Opportunities and challenges," *Academy of Management Journal,* vol. 50, no. 1, pp. 25-32, 2007.

[34] N. King, C. Cassell, and G. Symon, "Using templates in the thematic analysis of texts," in *Essential Guide to Qualitative Methods in Organizational Research*, C. Cassell and G. Symon Eds. London, UK: Sage, 2004, pp. 256-270.

[35] R. M. Grant, "Toward a knowledge-based theory of the firm," *Strategic Management Journal,* vol. 17, no. S2, pp. 109-122, 1996.

[36] M.-A. Storey and A. Zagalsky, "Disrupting developer productivity one bot at a time," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering,* 2016: ACM, pp. 928-931.

[37] T. E. Fægri, T. Dybå, and T. Dingsøyr, "Introducing knowledge redundancy practice in software development: Experiences with job rotation in support work," *Information and Software Technology,* vol. 52, no. 10, pp. 1118-1132, 2010.

[38] N. P. Repenning, "Understanding fire fighting in new product development," *Journal of Product Innovation Management,* vol. 18, no. 5, pp. 285-300, 2001.