Knowledge Retention Challenges in Information Systems Development Teams: A Revelatory Story From Developers in New Zealand

Yi-Te Chiu, Victoria University of Wellington, New Zealand*
Kristijan Mirkovski, Deakin University, Australia
Jocelyn Cranefield, Victoria University of Wellington, New Zealand
Shruthi Shankar, Victoria University of Wellington, New Zealand

ABSTRACT

Information systems development (ISD) is an integral part of organizational agility in today’s competitive business environment. High turnover, agile ways of working, and fluid work environments pose challenges for ISD. This paper explores the erosion of knowledge retention (KR) arising from ISD staff churn in a New Zealand-based financial organization in the aftermath of a major earthquake. In this exploratory study, the authors develop a causal model of KR in the ISD context, which articulates the challenges to and consequences of ineffective KR at the routine and exiting stages of KR. The model identifies four challenges—coordination complexity, insufficient resources for knowledge retention, insufficient attention to knowledge retention, and slow staff replacement and handover processes—that can affect the loss of ISD knowledge when routine and exiting KR fall into disarray. This study also reveals that role stress and reduced ISD agility reinforce the cycle of knowledge loss.

KEYWORDS

Agility, Banking Industry, Information Systems Development Professionals, Knowledge Loss, Revelatory Case Study, Role Stress

INTRODUCTION

The information technology (IT) industry has been at war for recruiting and retaining talent over the decades. The median employee tenure in the IT industry is significantly lower than many other global industries, such as the automotive, pharmaceutical, and telecommunications industries (PayScale, 2020). It ranges from 1.1 years at Amazon.com Inc. and Google Inc. to 6.4 and 7.2 years in IBM Inc. and Xerox Corporation, respectively (PayScale, 2020). Unsurprisingly, the high turnover rates in the IT industry have been attributed to the working environment, including workload amongst project members, variance in the working hours of project members, and imbalance in the working hours of project members (Bao et al., 2017). The transformation toward agile ways of working further intensifies the uncertainty of work environments. ING reduced its workforce by 25% during agile transformation (Kerr et al., 2018). Together these factors contribute to the loss of organizational knowledge that is embedded in departing employees.
In the face of the pressing need to remain digitally agile in an ultra-competitive business environment, IT organizations are likely to experience financial losses associated with the erosion of competitive advantage and poor organizational performance when critical knowledge is not retained (Daghfous et al., 2013; Harden et al., 2018). Knowledge retention (KR) is a management imperative for many IT organizations, especially for ones facing a greying IT workforce (Gonzalez, 2016) and/or those that rely on external labor and consultancy markets to fill talent shortages (Chaudhuri et al., 2018). KR is concerned with managing knowledge embedded in individuals and their relationships (Liebowitz, 2008; Martins & Meyer, 2012) and seeks to tackle knowledge loss (KL) challenges when knowledge is not captured, and knowledge workers leave organizations (Durst & Zieba, 2019; Huber, 1991). KR continues to be a major concern in information systems development (ISD). ISD knowledge is often not properly documented (Batra et al., 2011; Remus, 2012). Moreover, it is not uncommon for outgoing IT experts to be submitted to rushed exit interviews and for newcomers to be bewildered by unstructured handovers (Coombs, 2009). Losing information systems development (ISD) knowledge prevents IT personnel and/or contractors from delivering IT-enabled value (Lin et al., 2016) and cripples IT-dependent organizational agility (Fink & Neumann, 2007).

Building on risk management literature, Jennex (2014) suggests assessing the likelihood and impact of KL risks, formulating a KR plan, and monitoring whether risk responses are effective. Existing literature recommends various strategies to address risks of KL. Research has shown that knowledge management (KM) governance, talent management practices, electronic repositories, and KM routines embedded in ISD methods can address KL risks (Korimbocus et al., 2018; Kotlarsky et al., 2014; Levallet & Chan, 2019; Lin et al., 2016; Pflügler et al., 2018). Nevertheless, in a dynamic environment where the structure and the composition of the organization change constantly, sustaining KR remains elusive, as is evident in open source projects (Rashid et al., 2019) and software organizations (Alahyari et al., 2019).

Inspired by the learning from failure literature (Dahlin et al., 2018), we contend that studying a context where multiple adverse events lead to failed KR is of value. An understanding of the relationships of the causes and process can shed light on future design of KR in a dynamic environment. Since KR a complex KM process, ineffectiveness is not generally attributed to one single factor but by cumulative missteps. This exploratory study aims to disentangle interconnected relationships among key challenges to KR in ISD, and their impacts and consequences in IT organizations. Based on an in-depth case study of an IT unit in a financial organization (Pēke) from the banking industry in New Zealand, the authors developed a conceptual model that identifies (1) the factors hindering the performance of KR in organizations, (2) the nature of the KR process, and (3) individual and organizational consequences associated with ineffective KR. This study contributes to the literature on KR in the ISD context by proposing a novel practice-based conceptualization of the challenges that contribute to KL when routine and exiting KR practices are dysfunctional. The loss of ISD knowledge results in increased role stress of IT professionals and reduced organizational ISD agility. Underlying these insights is a dynamic and multilevel view of KR that considers individual workload, resource management, hiring process, management attention, and organizational structures. By understanding the KR challenges shaped by the action of diverse actors and organizational structures, academics and practitioners can learn ways of recognizing and steering around a cycle of KL.

The remainder of this paper is structured as follows. First, the literature on KR and KL in the ISD context is reviewed to position this study. Then details about the research methodology and the selected case are provided. Afterwards, the results of this study and its emergent model are presented. Lastly, the contributions and limitations of this research are discussed.

**LITERATURE REVIEW**

KR is concerned with preserving and maintaining the knowledge that is embedded in individuals and their relationships with others, and is used by organizations to help them cope with challenges
arising from the exit of employees (Liebowitz, 2008; Martins & Meyer, 2012). KR 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” (Levallet & Chan, 2019, p. 178). While KM strategies have been proposed to enable KR, including IT-oriented and people-oriented approaches, KL often becomes inevitable when knowledge is not retained properly or is difficult to reuse (Levallet & Chan, 2019; Martins & Meyer, 2012; Massingham, 2018). This can be attributed in part to the hurdles of knowledge conversion between tacit and explicit knowledge (Nonaka, 1994; Nonaka & Von Krogh, 2009) and to the reduced knowledge sharing (socialization) opportunities that arise from the loss of knowledgeable staff.

**Success of Knowledge Retention**

Individuals, organizations, and IT can contribute to success of KR. At the individual level, research shows that intrinsic motivation (Kankanhalli et al., 2005) and extrinsic motivation (Bairi et al., 2011) increases KM activities. For example, in cases when sharing knowledge elevates one’s power and reinforces one’s job security, individuals tend to have positive attitudes toward KM behaviors (Wasko & Faraj, 2005), resulting in KR (Aggestam et al., 2014). With regard to organizational factors, embedding KM practices in organization routines can sustain KR (Daghfous et al., 2013). Repetitive actions in creating documents, conducting expert interviews and holding workshops ensure knowledge flow and retention (Shankar et al., 2013). Moreover, organizational processes (i.e., information codification, information personalization, and electronic communication) contribute to organizational memory (Fiedler & Welpe, 2010). Organizational structures can reinforce organizational processes through standardization (e.g., centralized hiring processes and rules and policies to govern collaboration) (Fiedler & Welpe, 2010) or specialization, such as specialist roles and responsibilities (Fiedler & Welpe, 2010; Shankar et al., 2013). Besides formal structures and processes, organizational culture enhances motivation and shapes positive attitudes (Alavi & Leidner, 2001; Liebowitz, 2008). Leaders foster KM-centric culture (DeLong & Fahey, 2000) and also raise the awareness of what knowledge should be retained and why it is important (DeLong & Storey, 2004; Martins & Meyer, 2012). More specifically, leaders who set a strategic priority for KR and engage in KR initiatives inspire and stimulate KR (Gelard et al., 2014).

IT-enabled knowledge management systems (KMS) underlie conscious and automatic KR, from knowledge capture, storage to knowledge retrieval (Levallet & Chan, 2019). KMS come in the form of information systems that support collaboration, knowledge curation, search, and documentation (Lin et al., 2016). The realization of KMS value relies on system quality and information quality (Lin et al., 2016; Shankar et al., 2013). Resistance toward KMS cannot help KR, but instead aggravates KL (Levallet & Chan, 2016). Having a proper KMS in place and a supporting organizational environment helps the organization to effectively develop and use innate experiential knowledge for the benefit of the organization as a whole and all its stakeholders (Robinson & Ensign, 2009). However, there are limitations to what can be achieved from a codification-based approach to KM; peer-to-peer socialization within communities is seen as critical to the success or failure of knowledge retention and transfer, particularly in project environments (Bresnen et al., 2003).

**KR in the Context of ISD**

KR provides the necessary conditions to develop IS-enabled products and services by incorporating prior ISD knowledge and experience (de Souza Bermejo et al., 2016). ISD knowledge includes technical knowledge related to the IS applications and their underlying technologies, as well as organizational knowledge, including processes and structures (Livari et al., 2004). In theory, technical and organizational knowledge, such as architectures, databases, and business rules, is explicit and easily captured and retained (Damian, 2007). Careful documentation and configuration management can ensure critical knowledge is retained (Liebowitz, 2008). However, in practice, for the sake of convenience and speed, explicit technical and organizational knowledge is often inadequately
recorded, leading to so-called documentation debt (Alves et al., 2016; Rios et al., 2020). Further, much ISD knowledge is tacit and experiential due to the complex, abstract, and context-dependent nature of ISD work. For instance, clients’ needs are typically implicit and volatile (Kankanhalli et al., 2003) and thus require a substantial amount of interaction to be understood (Chakraborty et al., 2010). The content of tacit knowledge can also be technical in nature and reside in both individuals and teams (Ryan & O’connor, 2009). For example, local coding conventions and design practices often reside among experienced programmers and are difficult to transfer to newcomers (Rus & Lindvall, 2002). Even when experienced programmers express their willingness to share knowledge, incompatible programming mental models may prevent them from understanding one another’s coding practices (Shaft & Vessey, 2006). Further, KR can be challenging in the team environment where team members should understand not only who possesses what specialized knowledge (He et al., 2007) but also how to coordinate fluidly (Strode et al., 2012). The prevalent adoption of agile methodologies makes retaining ISD knowledge challenging (Behutiye et al., 2017) because, unlike traditional ISD methodologies, collaboration and working software are valued more highly than processes and documentation (Beck et al., 2001).

Consequences of Ineffective Knowledge Retention

Empirical research on the outcomes of KR for IT professionals is sparse. Recent work, such as the context-emergent turnover theory, suggests that collective turnover, conceptualized as inflow and outflow of knowledge and talent in an organization, alters the composition of human capital over time (Nyberg & Ployhart, 2013). In some circumstances, collective turnover has been found to reduce productivity and customer satisfaction (Bairi et al., 2011; Hancock et al., 2013). It also leads to coordination friction, especially occurring when experts leave, lowering task performance (Hausknecht & Holwerda, 2013; Summers et al., 2012). Besides a loss of task-related knowledge, collective turnover means a loss of social knowledge and lower organizational cohesion (Mowday et al., 2013), which reduces firm performance (Daghfous et al., 2013; Jiang et al., 2012). The consequence of turnover is particularly grave if the work environment involves a substantial amount of tacit knowledge (Eckardt et al., 2014), as is the case in ISD.

KL, an immediate consequence of ineffective KR, is often detrimental to organization performance (Holan & Phillips, 2004). For example, Massingham (2018) found that KL has negative impacts on organizational problems, including low productivity (morale), strategic misalignment of the workforce (capability gaps), resource cuts (stakeholders unhappy with performance), decreased work quantity and quality (inexperienced employees), work outputs not being used (customers mistrust), longer time to competence (learning cost), and slow task completion (increased search cycle time). As well, KL has largely contributed to increasing the risk associated with work activities, decreasing organizational capacity to manage risks, and reducing organizational knowledge resources.

RESEARCH OPPORTUNITY

Existing research examines the factors that enable KR. While success and challenges share similar characteristics, following what is prescribed in the success literature does not always lead to success, when so-called spurious failures occur (Dahlin et al., 2018). This can be attributed to the complexity of causes and unique contextual factors (Dwivedi et al., 2015; Klein & Myers, 1999). We contend that learning from failures can help to disentangle what causes failures in KR and elucidate how such failure occurs. Moreover, given the shift of ISD toward agile and dispersed development, a better understanding of how KR occurs in this context will add value. Such understanding may help prevent organizations from losing agility and aid in their realization of the value of IT (Baham & Hirschheim, 2021). To address these pressing needs, this study aims to reveal how challenges to KR in the context of ISD result in KL.
RESEARCH METHODOLOGY

The authors selected to study KR for ISD professionals in a financial institute. Owing to the significant dependence of ISD professionals on ISD knowledge to deliver organizational value (Iivari et al., 2004; Lin et al., 2016), the setting is suitable to investigate the challenges of KR and to examine the intertwined relationship between ISD professionals and the organizational context in which KR practices (and/or the lack of these) occurs.

The authors used an interpretivist case study strategy combining inductive and deductive analysis of data to develop a theoretical framework (Eisenhardt & Graebner, 2007) that outlines the key challenges to KR in ISD teams. Case study method is considered suitable for studying a real-life phenomenon in depth, when the focus is on contemporary events, and when there is limited knowledge about the phenomenon under study (Benbasat et al., 1987).

All the data was collected from the development unit in the IT department of a financial organization (Pêke) in the banking sector in New Zealand. This organization was purposively selected as a revelatory and extreme case (Eisenhardt & Graebner, 2007) because of organizational volatility (see more discussion in the next section). It met Yin’s criteria for a single revelatory case as it offered the opportunity to reveal insights about a phenomenon that had not previously been explored in depth (Yin, 2011). Pêke represented a rich and powerful case for the topic because they have been through significant KR challenges. It served as a crucible to understand challenges that have been simultaneously faced by many ISD professionals – fast delivery, fluid organizational structures, dispersed collaboration, and unexpected changes. and that impact on KR. The organization was an ideal revelatory case where researchers observed and analyzed phenomena that are less accessible (Yin, 2011). The focus was put on understanding the process of unintentional KL in the context of ISD in this organization, and its impacts.

Case Background

Pêke offers its customers a range of personal, business, and international financial services. IT has continuously played an important role in Pêke, enabling the provision of convenient services through a diverse range of digital channels and platforms. It was essential for the project teams to work together to implement effective IT solutions, which involved (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 the study, there had been a significant churn of resources at Pêke’s IT department. For many years prior to the study, the IT department had been composed of permanent and contract developer positions. A significant number of contract developers who had been recruited on a short-term contract basis had ended up working for 3 to 5 years, leading to an increasing dependence by Pêke on their expertise. However, not long before this study, Pêke’s senior management had terminated 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 had been difficult to recruit developers with the required skillset.

Following the downsizing, most ISD developers were new to Pêke and had less than a year of experience in their present roles. As long-term contract developers left, they had taken the knowledge and expertise that they had built up over time. The loss of experienced developers meant that the ISD teams had lost a lot of critical information around developing, supporting, and delivering IS solutions. Furthermore, this situation of KL was compounded after a severe earthquake of magnitude 7.8 Mw shook much of New Zealand in late 2016. Prior to the earthquake, all the project teams had been co-located in one city; however, after the earthquake, the teams were rapidly dispersed across multiple locations in different regions of New Zealand. As a result, interaction and collaboration in project teams and across the IT department became increasingly difficult. The dispersal of 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 relied principally on email and instant messaging for inter-unit collaboration and coordination.

DATA COLLECTION AND ANALYSIS

Evidence was gathered from multiple sources—interviews, company documents, and observation notes—to identify factors hindering KR and the impacts of KL. Six in-depth semi-structured interviews were conducted in August and September 2017, lasting between 40 minutes to 1 hour, with two inexperienced developers (Developers A and B) and four experienced developers (Developers C, D, E, and F) (see Table 1). Both experienced and inexperienced project members were interviewed in order to develop an in-depth understanding of the KR challenges and barriers associated with departing and new developers and ISD projects in general. Interviewees were asked to tell their stories about what, when, how ISD knowledge was shared in their projects. Sample questions included, “What kind of challenges have you observed the developers in your team or teams you interact with face when critical/key developers leave?”, “What do you think about the current knowledge retention process between developers when someone leaves or joins? Is it effective? Why or why not?”, and “When you initially started this role, what type of information did you get from your teammates?”. The total interview duration was approximately 5 hours, comprising approximately 40,000 words over 65 pages. In addition to the interview data, the authors also wrote post-interview field reflection notes to capture their interview impressions while they were fresh as well as recorded observation notes from the several informal conversations with the respondents, which resulted in a total of 5 pages (approximately 3,000 words) of notes. Moreover, the authors used company documents, such as the official organizational website, to better understand official organizational processes and perspectives. Altogether these sources of evidence provided us with an in-depth understanding of Pēke’s ISD teams and the challenges to KR that they were facing.

<table>
<thead>
<tr>
<th>Interviewees</th>
<th>Working Experience</th>
<th>Job Role and ISD Team Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer A (Female)</td>
<td>4 months</td>
<td>A full-stack developer; Member of a small development team composed of relatively new members (i.e., less than 2 years’ service).</td>
</tr>
<tr>
<td>Developer B (Male)</td>
<td>8 months</td>
<td>Technical lead of the team; Member of a small development team composed of relatively new members (i.e., less than 2 years’ service).</td>
</tr>
<tr>
<td>Developer C (Male)</td>
<td>5 years</td>
<td>Technical lead of the team; Member of a medium-sized development team composed of relatively experienced members (i.e., more than 5 years’ service).</td>
</tr>
<tr>
<td>Developer D (Male)</td>
<td>9.5 years</td>
<td>A full-stack developer; Works alone, without membership into any team. The only developer who supports several IT applications.</td>
</tr>
<tr>
<td>Developer E (Male)</td>
<td>5 years</td>
<td>Leads technical delivery; Interacts with several development teams of varying sizes that are composed of new and experienced members.</td>
</tr>
<tr>
<td>Developer F (Female)</td>
<td>10.5 years</td>
<td>Leads system analysis and design; Interacts with several development teams of varying sizes that are composed of new and experienced members.</td>
</tr>
</tbody>
</table>

All the gathered information was transcribed, including interviews and handwritten observation notes. The researchers then adopted the template analysis technique (King et al., 2004) for a thematic
analysis using NVivo 10. The thematic analysis started with the interview data and then the observation notes. An initial coding template was developed based on the interview data, which was further refined and triangulated based on the analysis of the observation notes. For example, “learning difficulty for onboarding ‘developers’” and “‘reinventing the wheel’” were two separate grounded codes in the initial coding template; however, after confirming that the prolonged learning for new developers was a consequence of new developers’ necessity to reinvent ISD knowledge from the observational notes, the researchers decided to merge these two codes into one: “prolonged ISD learning for new developers.”

The researchers took a mixed approach to the thematic analysis with both inductive and deductive coding. The first round of analysis involved inductive coding, and the second round of analysis involved deductive coding based on applying existing themes and concepts from the literature. The analysis was sensitized to an extent by knowledge of the knowledge-based theory of the firm (Grant, 1996), but the researchers allowed codes to naturally emerge from the data (i.e., inductive or grounded coding). For instance, emerging codes, such as “lack of confidence for new role”, “lack of adequate understanding of the IS applications or systems”, “difficulty of finding experienced developers with appropriate ISD knowledge”, and “prolonged ISD learning for new developers”, were developed inductively. The researchers returned to the literature after the inductive round of coding, when stress was identified as a consequence of KL. Then a deductive round of coding based on role stress theory was conducted to analyze the exact nature of this stress and categorize the emerging codes into a role stress theme. The authors constantly modified codes throughout the analysis process based on their usefulness and suitability, which resulted in modifying several themes. The final coding template consisted of five main themes and ten sub-themes related to the challenges of routine and exiting KR, KL, and individual and organizational consequences of KL, as shown in Table 2.

**FINDINGS**

The thematic analysis led to the development of a conceptual model that depicts the factors that conspired to prevent the performance of KR practices, and the associated outcomes of KL in the ISD team context (see Figure 1). Two distinct conceptualizations of KR were identified: (a) routine KR and (b) exiting KR. Routine KR refers to those activities that involved capturing and retaining ISD knowledge on a regular basis. Exiting KR refers to the transfer and storage of knowledge held by those departing from the organization, involving activities such as handover practices, hiring practices, as well as knowledge documentation and archiving practices. At Pēke, all these 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. Furthermore, these employees typically approached these problems with a problem-solving orientation rather than a knowledge-sharing orientation. Four key challenges to the performance of KR were also identified. These were coordination complexity, limited resources, limited attention to KR, and slow staff replacement and handover.

Another key finding from the analysis was that the identified challenges tended to have initial consequences at the individual level in the form of role stress (role ambiguity and role overload). However, the deleterious outcomes of KL on individuals in the ISD project teams eventually combined to create significant second-order team-level outcomes: reduced ISD agility, slower development and delivery, and lower quality (as shown in Figure 1). The conceptual model summarizes the causal relationship between the identified challenges, the diminishing of KR activities, and the consequences for individuals and the organization. These findings are outlined in detail in the following sections.
Coordination Complexity

Coordination in ISD is concerned with the management of knowledge and skill interdependencies (Faraj & Sproull, 2000) and task interdependencies (Brandon & Hollingshead, 2004). The authors observed that the complexity of coordination in Pēke arose from complex interaction patterns due to changes in organization structure and task allocation. As recorded in the observational notes, Pēke’s ISD teams were dispersed in two geographical locations in New Zealand due to the high magnitude earthquake in late 2016, which damaged the original headquarters of the IT department. Hence, the existing geographical and organizational dispersion impeded routine KR. Developer E stated that “After the earthquake, we are scattered and literally placed in different buildings right, and it’s not easy and it’s very difficult at the moment to work together, it’s challenging.”

The multiplication of dispersed work sites, after the major earthquake, contributed to more complex interactions and communication between developers. Developers needed to know when and how to work collaboratively and share knowledge. In addition to geographical dispersion, organizational dispersion added another layer of complexity. Multiple functional teams, including
development and operations teams, needed to find ways of working together for system integrations when implementing new or updating IT functionalities, which hindered routine knowledge activities. Even within the same unit, developers mostly worked independently rather than collaboratively as per the observation notes.

The complexity of coordination further increased as a result of fragmented task allocation. Developers in the same unit worked on different areas of a single application. Hence, each developer had a focus area that they developed and supported, rather than having knowledge of the entire application. Since the systems at Pēke were tightly coupled, developers in a team had to develop technical and organizational knowledge to complete tasks via interpersonal communication. Such communication often required experts who were external to the team:

*The biggest problem in IT is communicating difficulty between two tasks, so two tasks may look relatively the same, but you know that one could be 100 times harder than one and to an end user it might seem exactly the same difficulty, so that kind of stuff sharing across the organizational level and sharing across the team to say.* (Developer B)

Developer F further added that such complex coordination required interpersonal communications, which posed challenges to newcomers: “The way we get things done around here is by having contacts, and it’s more challenging now for new people.”

Ad-hoc, sporadic communication with external team members, along with dispersion, further compounded routine KR. The recent downsizing of teams, which was recorded in the observation notes, exacerbated the KR issue. In situations where there was only one developer who held both development and maintenance roles for several IT, ISD knowledge was barely retained. Knowledge about those critical systems was often only held by a single individual. Because such developers did not have the privilege of working with others as part of a team, they lacked anyone to share or transfer their knowledge to:

*Again, it’s mainly around, in my team, not knowing what processes we are supporting, it’s really scary when you think about it, because when they look at the code, they get a picture but there might be other services that are calling it so people take things lightly when they should take it lightly, and that’s the risk […] because they don’t have prior knowledge of that functionality and we don’t share knowledge within the team.* (Developer F)
Insufficient Attention to KR

Insufficient attention to KR from management had been a major issue. KR was not considered as part of developers’ roles and responsibilities as there was little prioritization for KR and encouragement for sharing from management. As recorded in the observational notes, KR was not the strategic priority of Pēke. In fact, people who had previously facilitated KM were disbanded. Without the push from management for KM practices, developers were not motivated to transfer their accumulated knowledge to others:

I think everyone agrees knowledge management and sharing is a great thing to do, but getting it prioritized enough to actually do the work is not something that is actively worked on...I think there is a desire for it [KR] from management, I don’t think it’s happening a lot, but it has been talked about. (Developer E)

A developer expressed concerns about a lack of attention to routine KR at Pēke:

There is no process for sharing, so the sharing we have, people leaving and people joining is not quite robust enough, and it doesn’t have the right information captured to feed into it so it’s ineffective. (Developer F)

Besides management’s claims that KM was important for teams’ information flows, developers felt that hardly any constructive actions had been taken to encourage KR within the teams. A developer described his impressions:

Management 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. (Developer C)

Over time, developers developed DIY attitudes, orienting towards solving problems on their own. Developer B indicated that “from 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.” Developer C reflected on developers working in silos: “We might be working on similar things at the same time, there might be overlap, and it is caused by not having communication.”

A lack of incentives and encouragement reinforced knowledge hoarding behaviors. Developers tried to remain the key individuals or experts for the applications they maintained. In particular, senior developers who solely supported legacy systems had hoarding tendencies. They were reluctant to share their knowledge, so as to keep their positions secure. Being the only individuals who knew how the system worked meant that they had control and could dictate the development, deployment and support of the system.

The sort of knowledge that I have, I could be useful to the bigger team, if I shared that knowledge, but some of it is the fact that I like to hold on to the knowledge, it’s probably a bad thing, I don’t know I guess part of being key person, I’m not one to give up knowledge that easily, I guess. Because I’ve worked for so long by myself, I prefer that because I can just get on with my thing and not being pressured by anyone else working with me. (Developer D)

A developer indicated a need for attention to KR, even starting from small:

Something like release management to different locations and how we build and release, if someone gets 10 mins time then they should share that, it will be knowledge among all team members and will be a good session to learn and understand. It should be done but this is not happening. (Developer A)
Insufficient Resources to KR

Workload pressure and lack of time hindered KR in project teams. Respondents pointed out that, due to resource churn in the IT department, existing developers had an increased workload, due to trying to fulfil the responsibilities of developers who had left:

For years I’ve been sort of you know getting onto management about that [KR] there’s been absolutely no support whatsoever. No resources allocated, and when a resource does get allocated, you know there’s a change of plan. (Developer D)

Lean operations in Pēke impaired routine KR:

I think resource constraints is the biggest challenge in our team, we don’t have enough people to do the project work so spending time transferring knowledge and providing documentation is a premium that we don’t have. (Developer B)

Additionally, with the volume of project work prioritized by management as recorded in the observation notes, developers did not have sufficient time for documenting or sharing knowledge with their peers. Developer C said that, “IT teams basically run, run, run every day and nothing slows down to talk to each other and document things.” A developer was concerned about limited resources for KR.

I don’t have the luxury of having a team of people where we all know our bits. There’s no one left, if there was someone here that I had done a knowledge transfer to, then it would’ve been easier. (Developer D)

Challenges ofExiting KR Practices

Slow Staff Replacement and Handover Processes

As per the observational notes, the hiring process for permanent positions took excessive time due to the required approvals from various organizational levels, which added to the rushed handover process. New developers usually joined only a few days before departing developers left, which did not give sufficient time for adequate knowledge transfer:

It takes a very long time to get approvals to hire a new person, it’s not straight away we normally don’t hire people before the previous person leaves, that’s a bigger problem as well, so it’s usually after 3 or 4 weeks we get a new person, and by that time there is no one to do handover. We are not really organized when it comes to these things… [o]ne of the worst handovers I had that was given to me was two hours before the guy left the company. (Developer F)

Related to insufficient resources for KR, for developers who were about to leave, their workload during their notice period increased, as they were expected to quickly complete all the pending tasks before leaving. They still did their daily work right through to the point when they left. With the increased workload, the departing developers found it difficult to share knowledge before leaving the company. Moreover, developers didn’t invest adequate time or effort during their notice period as knowledge transfer was not prioritized over project work:

They only have 4 weeks leave notice or shorter but they are still expected to continue with their daily activities in terms of writing and delivering code and doing work right through to the point where they leave, so trying to find time in their schedules to either document or time to handover to a new
staff member, who may not even have started yet, those sorts of process difficulties make it hard. (Developer E)

As recorded in the post-interview field reflection notes, new and existing developers found the handover process to be rushed and dense as many new developers usually had only one or fewer hours of formal handover during the induction. A developer expressed frustrations for rushed handover:

*I just got a one-hour session from the person who was leaving, so he just showed me the code structure and database code. Just a one-hour overview of how the code works and what technology we are using. I also got some reference documents and links to start with, but I feel like it was just a basic level.* (Developer A)

New developers were 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 the following analogy for handover between developers:

*It’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 French neuroscientist.* (Developer E)

It is also noted that the handover process provided only basic ISD knowledge that was not useful for solving complex issues spanning multiple systems and involving multiple business users. A developer indicated that:

*From the initial training … we didn’t have any visual documentation on where things are or for incidents is a good example. There was no documentation on how we solved them so then we had to rework it out when they came in.* (Developer B)

The handover process was not structured, and much crucial information was missed depending on the departing developer’s time and personal motivation. As departing developers were usually under workload and time pressure, they were too busy to help or adequately transfer knowledge to new developers. Such informal knowledge transfers provided only basic and high-level information about IT.

The 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. (Developer E)

**Insufficient Attention to KR**

As per the observation notes, new developers acknowledged that there was a lack of any formal KR policies or processes for capturing, sharing, and transferring accumulated work knowledge when they joined the team. IT managers never prioritized KR as there was always a higher priority for developers to focus on. This was a significant issue when current developers were departing, and new developers were inducted. KR was usually not planned effectively in which departing developers had a four-week notice period to hand over their responsibilities to the new developers. New developers were not formally trained to capture and share their knowledge, nor to utilize any mechanisms or technology:

*We’ve had a new developer start, she is an experienced developer, and she was expected to take on [from]...the person who left, taking on their virtual machine for example, but that is not an easy thing to do with permissions and access controls etc. Then I suppose, as a good candidate, she just jumped in and figured it all out, but was not an awful lot of mentorship or handover from what I saw. Which
is disappointing because from my perspective, what would be better would be for that person to start with as much advantage they can get. (Developer E)

New developers acknowledged that there was a lack of useful, relevant, and up-to-date documentation created by departing developers for them to refer to. Although numerous documents were available on the intranet share-point sites, most of them were not considered to be current or relevant anymore. There was a lot of documented information that was out of date as the applications had changed significantly. Documentation regarding business and functional requirements of the project and architecture design were documented before starting development for approvals and signoffs. However, due to a lack of archiving process and prioritization for documentation after commencing development, these documents were not updated regularly as the project evolved. Developer A stated 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 a lack of documentation created by departing developers, particularly around the high-level architecture of IT and their integrating components. Also, information was not documented regularly for support or maintenance of the applications. Thus, documentation was limited, and when it was required, it was produced with insufficient information. Often new developers were not aware of documentation created by developers who had departed from working on their application or project, as they were not told about it. Additionally, there were a vast number of documents on Pēke’s SharePoint site that were not categorized or structured properly for easy access. The documentation was stored randomly with different teams and developers following their own practices for documentation:

*We might be working on similar things at the same time; there might be overlap, and it’s caused by not having communication. The nature of our team is such that we don’t have resources to be duplicating things with the workload.* (Developer F)

**Role Stress as Individual Level Consequences of Ineffective KR Practices**

**Role Ambiguity**

Developers lacked confidence in their new role because they did not have a complete understanding of IS that they are required to support. They were often worried and uncomfortable making changes to the codebase and deploying those changes as they were uncertain about their impacts on the business processes and other systems. A developer commented that:

*If you lose senior developers, 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.* (Developer C)

New developers were thrown into the deep end, where they were expected to pick up responsibilities without adequate documentation and training from the developer they were replacing. They pointed out that they struggled with locating relevant information and individuals within the organization that could help them solve an issue faster. Significant time was spent on searching for knowledge which in turn prevented new developers from acquiring the necessary skills to efficiently and effectively work on their given tasks:

*Although my team member helped, I ask them individually by myself to gain the knowledge, because sometimes people are busy with other stuff and don’t get time, so it’s understandable, but in that way, if you have the process planned then it should help. There are some items I should be knowing,*
because sometimes I can do things but I’m not aware what I should do. Like if something was told to me, then I could’ve done it. (Developer A)

As recorded in the post-interview field reflection notes, tracking down key developers with appropriate knowledge about a specific feature was one of the main difficulties for newcomers. When developers required additional information to successfully implement and deploy IS, they often had challenges attaining relevant and correct information on their own as most of the documentation was out of date. Due to knowledge silos, information about a feature was usually trapped in one individual developer’s mind, making it difficult to attain that information. New developers often felt lost when trying to find the right person with the right information and took a very long time approaching numerous individuals in this search process. A developer indicated that:

>You always need to go to a 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. (Developer F)

**Role Overload**

KL was a major concern among all the interviewees, as recorded in the post-interview field reflection notes. As experienced developers left the organization, the knowledge they had gained over the years regarding the development, integration, deployment, and maintenance of critical IS components was being lost. A valuable understanding of the systems, applications, and their business functions was not being retained within the organization when developers departed. Thus, critical technical and organizational knowledge was lost when developers left as they were not able to provide sufficient guidance to existing or new developers to acquire the knowledge needed to take over their responsibilities. A developer felt frustrated with the resulting lack of knowledge:

>I did not find any document, maybe the documents are not there or maybe the proper sessions for providing information like what is the full architecture of the project, how where the project stands among the other projects in [Peke], 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. (Developer A)

As per the observation notes, 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 is essential when implementing IT as developers need to recognize the impacts on business processes and dependent systems when updating a piece of code:

>What I got is how you set up your system and VM [virtual machines] and code, but what the project is and how database server and applications are working, not full details, but overview and architectural overview and server information overview and if you are dealing with different teams for getting the data, then what is the role and how frequently... the development understanding the business logic is important, we also need some other stuff like release management, deployment and all... so the full overview in different areas is still missing. (Developer A)

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:
Developers not knowing what processes we are supporting, it's really scary when you think about it, because when they look at the code they get a picture, but there might be other services that are calling it so people don’t understand that and they shouldn’t take it lightly as that’s a huge risk. It’s a struggle as well and takes some time for them to understand it. (Developer F)

Understanding the business context is crucial for developers to realize the business processes, functionalities, and requirements. A developer emphasized the importance of multiple aspects of knowledge required for ISD:

When 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. (Developer F)

Developers took 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 needed to dig through the codebase or search for a key person to figure out aspects of IT, which is often time-consuming and inefficient:

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 B)

Developers had also been asked to rebuild organizational knowledge required for systems, which is beyond their role requirements.

We 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. (Developer F)

Reduced ISD Agility as Organizational Level Consequences of Ineffective KR Practices

Slower Development and Delivery

As indicated in the observational notes, when existing and new developers struggled to cope with losing technical and domain expertise, the overall team morale was affected. With a lack of complete understanding of the systems and processes, developers felt uncomfortable picking up development tasks. The uncertainty and discomfort among the developers reduced the team morale as they took time to build confidence again as a team. Developer F felt that “when people join, they’re actually sometimes they feel isolated because they’re not getting that support, it’s really sad.”

With lower team morale and team members’ confidence, the team was not able to deliver IT as effectively and efficiently as they used to. As experienced developers left and new developers struggled to understand the systems and their broader context, the overall team’s capacity and effectiveness decreased. Hence, delays in IT project delivery were collectively noted as a critical consequence of reduced team capability and efficiency:

The direct impact is very slow working, mainly around time, you need to start doing all the rediscovery right, where we used to spend maybe 5-10 minutes, we spend days or weeks, we need to understand the whole end to end process before implementing things because we don’t want to break existing functionality as we go. (Developer F)
Quality and security were critical for Pēke as its information systems deal with sensitive or confidential data. As new and existing developers did not have a complete understanding of IT applications, they were not aware of the impact their code changes on other functionalities or systems downstream, which lower the quality of IT. A limited understanding of technical and organizational knowledge introduced defects:

If someone fixes a piece of code and they’re not aware that this can impact another thing also because they don’t understand they are related, then it will raise another defect, so at the end, the full cycle repeats, and it’s a waste of time. (Developer A)

Furthermore, the IT department did not deliver solutions effectively and efficiently as developers face challenges finding people for critical information and lack confidence in developing, integrating, supporting and delivery of projects:

When you hire people without banking experience, and when you have less resources available then it’s not really good. They can’t pick up things and run with it. They need support.” (Developer F)

Lower Quality

In one-developer teams, the developer was forced to provide support when they took long holidays as there were no other developers with even the slightest knowledge about the systems to support if required. Developer D indicated that, “being a key person, whenever I go on holiday, I’m still on call, and there is no backup or support.” Key personnel risk also led to concerns of KL regarding legacy systems making such key developers harder to replace. With the rapidly changing landscape of programming and technology, it has been difficult to recruit new developers with relevant skillsets to support legacy systems as many of the developers available in the current market would not have experience with obsolete technology.

Social connections and personal relationships were a fundamental aspect of the IT department culture where developers needed to establish networks and have relevant contacts within the IT department to be effective. As indicated in the post-interview field reflection notes, knowledge silos have led to a “shoulder-tapping” culture as it was 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 became difficult for new developers to work well without having any other access to obtain the relevant information. Thus, this largely contributed to the delivery of lower quality service in the ISD teams:

If we lost certain people in the team, there’s quite complex systems in the [financial institution] that are old and legacy, and we wouldn’t be able to support them properly. (Developer B)

Similarly, because developers were not fully aware of the implemented or existing features, they ended up re-inventing the wheel instead of reusing functionalities, resulting in re-implementation and lower quality of delivered service. This was because re-implementation was sometimes seen as easier and quicker than searching complex codebases or finding someone who could provide the information:

Probably 90% discussing with my colleagues and 10% looking at documents, but then again, I’ll talk to my colleague, and they’ll say “I don’t know how that works” and then reverse engineering and looking at documents, we don’t have a lot of documents particularly up to date so it’s a bit of a mess. (Developer B)
DISCUSSION

The findings illustrate the interaction of factors in an ISD organization that was trapped in a self-reinforcing cycle of KL. From a strategic choice perspective, this cycle can be seen as starting when an organization does not recognize the value of KR. Once critical knowledge slips away, KL puts pressure on newcomers who are unable to locate missing knowledge and need to reinvent the wheel, jeopardizing organizational IT capabilities. Particularly, newcomers in this study suffered from missing tacit ISD knowledge in relation to Peke’s IT architecture, which was inextricably intertwined with Peke’s organizational knowledge. Since IT personnel’s capabilities are crucial to ISD agility (Fink & Neumann, 2007), a lack of critical ISD knowledge prevents newcomers from developing their in-situ capabilities and collaborating with existing team members, leading to slower ISD agility. Research has shown that the performance gap between high and low ISD agility is significant—a high performer can roll out new digital options while maintaining a seven times lower change failure rate than low performers (Forsgren et al., 2017; Forsgren et al., 2020). This study illustrates how ineffective KR can diminish ISD agility.

The theoretical model identifies factors impacting KL, including the organizational structure dimension (i.e., coordination complexity and slow staff replacement and handover processes) and the managerial dimension (i.e., insufficient attention from management and insufficient resources). This study also reveals and classifies KR in two stages: routine and existing. Since critical ISD knowledge contains a large amount of tacit knowledge, it takes time to capture, transfer, and be reused. When organizations cannot follow routine KR (Levallet & Chan, 2019; Liebowitz, 2008), it is difficult to retain ISD knowledge when employees are about to leave. When organizations cannot break the KL cycle, IT professionals experience role stress as they attempt to deal with demands beyond their role description and rebuild lost ISD knowledge to accomplish their assigned goals. Furthermore, they are forced to do too much work in too little time. High role stress reinforces the cycle of KL as less time becomes available to seek and share knowledge. Considering the high fluidity of organizations nowadays, the authors believe that these inhibiting factors are prevalent. Organizations should pay special attention to their KR strategies and practices.

Implications for Theory

The theoretical implications of this study in relation to KR and KL are twofold. First, to the best of our knowledge, there are no prior studies that have identified individual role stress because of KL in the context of ISD. Previous studies show that role stress for IT professionals comes from work demand (Lim & Teo, 1999) and evolving technologies (Tarafdar et al., 2007). According to Lazarus’s transactional model of stress (Lazarus, 1995), stress arises from cognitive appraisal, including evaluating the work environment and whether one has capabilities to cope with the situation. This study demonstrates that knowledge loss at the organization level creates a more stressful work environment. The cycle of knowledge loss in the absence of retention appears to exacerbate role stress. The self-reinforcing cycle starts with a mix of factors, including insufficient organizational support/resources for KR, ineffective human resources management processes, and coordination complexity. A lack of routine and exiting KR leads to a diminished ISD knowledge base, which, due to the increased level of uncertainty, creates unclear work demands. Existing team members do not know if KR is part of their job. New team members feel confused about their ISD work. Then role overload occurs because they do not have the capacity to fulfill what they need to accomplish. Newcomers particularly suffer from role stress.

Second, this research stresses the strategic role of KR in ISD agility. The existing literature suggests that ISD agility can be built via strategies and mechanisms, such as the adoption of agile methodologies (e.g., Conboy, 2009; Conboy & Carroll, 2019), the design of adaptable IT infrastructure (e.g., Roberts & Grover, 2012), and strategic IT alignment (e.g., Tallon & Pinsonneault, 2011). Although knowledge capabilities underpin the development of agility, the focus has been put on
acquiring and integrating knowledge for decision-making in response to change (Huang et al., 2014; Park et al., 2017) as well as competence required for building components to achieve agility (Fink & Neumann, 2007; Kim et al., 2011). Put differently, the current theoretical understanding helps IT organizations obtain agility. However, little is known about sustaining agility. This study sheds new light from the knowledge retention perspective. Ineffective KR and knowledge loss can slow down ISD even when the organization possesses capabilities in sensing ISD-related changes and a flexible IT infrastructure. The KR issue is of relevance to the work environments characterized by high turnover, the adoption of agile approaches, and fluid work arrangements. Organizations that intend to sustain ISD agility should learn from errors and failures and avoid missteps.

In keeping with the Jennex-Olfman KM success model (Jennex & Olfman, 2006) and its revision (Jennex, 2020), we observe that Pēke’s limited capabilities in KM system quality, knowledge quality, and service quality constrained the organization in realizing the value of ISD and caused role stress. Our study extends this line of work by revealing the process through which KR failure can occur in the absence of KM success dimensions, via a recursive self-reinforcing process. It reveals how and why ISD organizations can become trapped in self-reinforcing cycles of KL. Our research also contributes to the learning from failures literature (Dahlin et al., 2018; Dwivedi et al., 2015) by portraying the failures that can be experienced by individuals and an organization in the context of failed KR in ISD. A revelatory case study vividly portrays a complex KR problem and provides developers and organizations with an opportunity to learn. Future research can take a step further and investigate how to break self-reinforcing cycles of KL.

**Implications for Practice**

It is suggested that organizations carefully devise a KR strategy and build KR into the routine and exiting stages. KR practices can be implemented through personalized approaches, such as mentoring, storytelling, and oral histories (Liebowitz, 2008), along with KM systems, such as the electronic community of practices and knowledge repositories (Kotlarsky et al., 2014). Recent advancement of intelligent software agents, such as Documentation Bots and DevOps Bots, shows potential to capture critical knowledge at the routine stage (Storey & Zagalsky, 2016). Routine KR practices reduce the pressure when there is limited time available for exiting KR. For instance, making routine KR align with performance reviews and 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 (Fiegri et al., 2010). Such routines can ease the process of knowledge identification and boost confidence in knowledge reuse. Exiting KR is more challenging, as departing experts are constrained by time and lack of motivation to transfer knowledge to organizational memory. IT managers and ISD team members may also have difficulty identifying what knowledge is critical to retain. Structured exit interviews, therefore, require a clear focus on KR and adequate time to consider these issues.

Second, the turnover intention of IT professionals is affected by role stressors (Calisir et al., 2011; Harden et al., 2018; Naidoo, 2018). This research discovered that ineffective KR is one of the causal factors for role stressors. Organizations must consider maintaining manageable role stress, especially for newcomers who do not have the resources to cope with stressors well. It is recommended that more resources should be invested to develop effective KR practices at the routine and existing stage. Managerial support and attention are also essential to support IT professionals.

Lastly, the findings point to a somewhat 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 it very difficult to escape the path-dependent trajectory of ineffective KR practices because these practices become self-reinforcing over time. It is likely that Pēke’s misfortunes had 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 KL trajectory.
Such strategic choice, if not complemented with appropriate practices to retain knowledge, can thus contribute to perpetual patterns of firefighting (Repenning, 2001).

Limitations and Opportunities

As this is an exploratory study of KR barriers and challenges in the ISD context, the findings have some limitations that provide opportunities for future research. While particularly revelatory, the strong contextualization is both a strength and limitation of the study. The generalizability of the findings to other settings could be limited as factors such as industry composition, infrastructure, or culture might play a role. The authors thus encourage researchers to test the generalizability of the model not only in other industries but also in other geographical regions and cultures. They can also extend attention to the contingencies, such as environment volatility, scarcity and munificence, organizational slack, and organizational turnover.

Due to the limited sample size of the current research, it would be useful to test the proposed model with larger samples to establish the validity of the model and further refine it. Large-scale surveys using confirmatory SEM, analysis of secondary data, or multiple case study can be considered. This also opens the opportunity to define and refine measures that are specific to the model, and that will support the key themes. That being said, a single case study based on revelatory accounts offers rich insights to explain the phenomenon that are often inaccessible and less accessible (Yin, 2011).

CONCLUSION

In this study, the authors 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 inductive and deductive thematic analysis of rich interview data, a practice-based conceptual model was developed 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. This study contributes to the literature on KR in the ISD context by introducing a novel practice-based conceptualization of KR comprising two stages and by explaining the self-reinforcing nature of the overall KL process in such a setting.

ACKNOWLEDGMENT

A short version of this manuscript was published in the Proceedings from Hawaii International Conference on System Sciences (HICSS) 2021. This version makes additional theoretical contributions regarding role stress and ISD agility, based on further analysis of the existing data.
REFERENCES


Yi-Te Chiu is a lecturer at the School of Information Management, Victoria University of Wellington, New Zealand. He holds a Ph.D. in Management from Smith School of Business, Queen’s University, Canada. Prior to his academic career, he worked in the software industry and helped organisations derive value from IS/T (Information System and Technology). His research revolves around emerging socio-technical opportunities and challenges in the digital work environments. At the individual level, his research concerns symbiotic relationships between workers and technologies, aiming to understand how workers adapt to emerging IS/T and how IS/T professionals can discover and deliver value through better IS/T development. At the enterprise level, his research aims to discover how to build and sustain IS/T-related organisational capabilities for digital transformation. He is also enthusiastic about applying IS/T to foster teaching, learning, and research environments.

Kristijan Mirkovski is a Senior Lecturer (Assistant Professor) in the Department of Information Systems and Business Analytics in Deakin University. He received his Ph.D. in Information Systems from the City University of Hong Kong in 2014 and have held various academic positions in Hong Kong, Australia, and New Zealand. His primary research focus includes ICT adoption and use, social media and commerce, supply chain management, alternative genres for communicating scholarly knowledge, and entrepreneurial and open innovation. His research has been published and accepted in Information Systems Journal, Supply Chain Management: An International Journal, Electronic Commerce Research, Internet Research, Information Technology and People, Information Systems Frontiers, International Journal of Logistics Management, IT Professional, and Supply Chain Quarterly. He has also presented his research at various national and international conferences, such as the International Conference on Information Systems, the Annual Meeting of the Academy of Management, the Americas Conference on Information Systems, the Pacific Asia Conference of Information Systems, and the Hawaii International Conference on System Sciences.

Jocelyn Cranefield is Senior Lecturer at New Zealand’s Victoria University of Wellington. Her research focuses on the intersection of information systems, human behavior, and management. Areas of interest include the impact of emerging IT and systems on work, leading digital transformation, and using social technologies to support knowledge management and professional change. Jocelyn’s research has been published in journals including JAIS, CAIS, and Knowledge and Process Management. She regularly presents at international information systems conferences such as ICIS, PACIS, ECIS, and HICSS, and is co-editor of two books. She enjoys working with industry practitioners and non-experts, drawing on prior experience in visitor experience design, government, publishing, and television directing.

Shruthi Shankar is a Senior Business Analyst and has worked predominantly in the banking industry. She completed her postgraduate studies at Victoria University of Wellington where she graduated with a Master of Information Management with Distinction.