1. Introduction

1.1. Cross-Border Collaboration in Offshored Software Development

To date, the way in which software is developed has undergone considerable changes. It is well known that successful software development projects require diverse team members with specialized knowledge pools to leverage each other's knowledge, solve problems and produce ideas collaboratively to accomplish software development tasks (e.g., Faraj and Sproull 2000; Levina 2005; Tiwana and Mclean 2003). While these specialized pools of knowledge are essential for software development, they create knowledge boundaries which impede cross-border communication and collaboration. Prior research emphasized that in order to realize the potential benefits of such knowledge heterogeneity, diverse team members need to address such knowledge boundaries (Guinan et al. 1998).

However, addressing knowledge boundaries is particularly difficult when clients outsource software development projects to external vendors. Indeed, the knowledge of clients and vendor employees is specialized in very different knowledge domains. Clients often create their software vision over several years, whereas vendor employees are often completely unfamiliar with the desired software product at the outset of a project, yet they have the technical knowledge to develop it (Tiwana 2004). Bridging knowledge boundaries is even more challenging when clients outsource software development projects to vendors situated in offshore regions such as India, China, the Philippines or Vietnam due to cost pressures and a lack of skilled labor (Kotlarsky et al. 2014; Oshri et al. 2012; Vlaar et al. 2008). While offshoring increasingly enables clients from industrialized countries to realize novel software visions without depending on the financial leverage of large investors (Gefen and Carmel 2008), many clients are unable to leverage the anticipated cost savings. A major reason why such benefits are often not realized is that projects result in unexpected extra costs. In particular, overly extensive knowledge sharing can offset the benefits of offshoring (Dibbern et al. 2008). In offshore settings, it is very challenging to collaborate across borders and develop shared understandings since clients and vendor employees are geographically and temporally separated (e.g., Kotlarsky et al. 2014; Vlaar et al. 2008). Hence, opportunities to communicate spontaneously and informally with each other are limited (Dibbern et al. 2008; Vlaar et al. 2008) and lack shared contexts (Carmel 1999). As a consequence, knowledge boundaries are not only prominent at the outset of offshored software projects, but are

likely to continuously challenge client-vendor collaboration throughout the project over longer periods of time (Levina and Vaast 2008).

To date, we have a limited understanding of how clients and vendor employees can collaborate across borders and address knowledge boundaries over time under such extreme conditions. Over the life of offshored software development projects, clients and vendor employees are likely to face different situations in which knowledge boundaries impede their cross-border collaboration. However, we have little knowledge on how they can address such knowledge boundaries in particular situations and over time in order to collaborate effectively. Hence, the overarching goal of this dissertation is to improve our understanding of how and why clients and vendor employees can successfully address different knowledge boundaries and leverage each other's knowledge specializations over the life of offshore-outsourced software development projects.

1.2. Using Boundary Objects to Collaborate across Borders

Today, it is well known that knowledge boundaries can be more effectively bridged when objects are used (e.g., Bechky 2003b; Carlile 2002; Majchrzak et al. 2012). In software development projects, the software prototype has become a common object of clients and vendor's practices mediating interactions between them (Floyd 1984), particularly when agile software development methodologies are applied (Boehm and Turner 2003). Collaboration scholars have increasingly taken a practice perspective to understand the ways people do things and engage in activities (e.g., Bechky 2003b; Carlile 2002; Majchrzak et al. 2012; Nicolini et al. 2012). Practices are defined as "recurrent, materially bounded and situated social action engaged in by members of a community" (Orlikowski 2002, p. 256). By investigating the daily activities of team members, it is possible to gain deep insights into how team members deal with the boundaries they face during their work, and capture the practices they regularly engage in (Orlikowski 2002). We know from prior literature that objects can be used in different ways to address knowledge boundaries (e.g., Carlile 2002; Majchrzak et al. 2012) and that depending on how they are used, objects may operate as *boundary objects* (Levina and Vaast 2005). Boundary objects are "both plastic enough to adapt to local needs and constraints of the several parties employing them, yet robust enough to maintain a common identity across sites" (Star and Griesemer 1989, p. 393). Prior research revealed that many different kinds of objects such as sketches, prototypes,

and designs can be useful boundary objects (e.g., Bechky 2003b; Carlile 2002; Ewenstein and Whyte 2009).

1.3. Overview of the Dissertation

The dissertation encompasses three separate studies concerned with the use of boundary objects during client-vendor interaction in cross-border software development. Each study has a distinct focus. The following paragraphs reveal how these three studies differ and build on each other. In particular, motivations, research goals and the gaps they address are outlined for each study and an overview of all studies is presented.

In the first study, the goal was to examine *how different practices help transform software prototypes into boundary objects to bridge different types of knowledge boundaries in offshore outsourcing software development projects.* While knowledge boundaries can impede client-vendor collaboration in software development projects, bridging such knowledge boundaries is particularly challenging when software projects are offshore-outsourced. Prior research has shown that objects can be critical for bridging knowledge boundaries when they operate as boundary objects. While many scholars borrowed arguments from boundary object theory (e.g., Bechky 2003b; Bergman et al. 2007; Carlile 2002), few authors have focused on how objects actually become boundary objects. In particular, the findings of prior scholars suffer from fragmentation and potential contradictions because a huge variety of different objects was studied in various contexts without systematically capturing the properties that enable objects to become boundary objects. In this study, we draw on boundary object theory, which proposes that certain boundary object properties need to emerge to qualify objects as boundary objects and that boundary objects are multidimensional and established in practice (Star 2010; Star and Griesemer 1989). Earlier scholars indicated that ongoing object use is vital for objects to operate as boundary objects (Levina and Vaast 2005). Yet, to the best of our knowledge, this study is the first to explore how software prototypes can become boundary objects by examining how the properties that transform objects into boundary objects may emerge through using software prototypes in different ways to bridge different types of knowledge boundaries, i.e., syntactic, semantic, and pragmatic (Carlile 2002).

In the second study, the goal was to understand *how and why practices with objects can bridge particular knowledge boundaries effectively and swiftly over time.* Prior scholars distinguished two alternative ways in which objects could be used to address

knowledge boundaries. While some scholars argued that team members need to engage in practices with objects that remove knowledge boundaries permanently (e.g., Bechky 2003b; Carlile 2002), others proposed practices with objects that enable swift cross-border collaboration (e.g., Kellogg et al. 2006; Majchrzak et al. 2012). To date, scholars examined these two alternative ways of using objects separately; yet, both ways of using objects are important to address knowledge boundaries in cross-border interactions. Prior offshoring literature showed that knowledge boundaries can continuously challenge client-vendor collaboration over longer periods of time (Levina and Vaast 2008); yet, investing heavily in knowledge sharing for removing knowledge boundaries permanently may offset the anticipated cost savings in offshoring (Dibbern et al. 2008). Thus, it might be desirable that clients and vendor employees use objects in ways such that knowledge boundaries are swiftly removed over time, without investing considerable time and effort. Thereby, we believe that clients and vendor employees face different situations in which they need to address knowledge boundaries that vary in their level of complexity. These different knowledge boundaries may need to be addressed in various ways. We recognize that objects may differ not only with regard to how they are used, but also with regard to how they embody knowledge. For example, objects may not embody up-to-date knowledge (e.g., Kellogg et al. 2006). However, to date, we know little about how software prototypes may be used to bridge particular knowledge boundaries with a given level of complexity swiftly and effectively during an offshored software development project. Thus, in this study, we explicitly capture the dynamics involved in bridging knowledge boundaries effectively and swiftly over a period of 10 months.

In the third study, the goal was to understand *how software prototypes and idea practices co-evolve such that clients and vendor employees could collaborate effectively and produce ideas collaboratively for improving the software product over the life of an offshored software development project.* To understand this emergent phenomenon unfolding through everyday interactions, we conducted a process-oriented study (Langley 1999) over a period of 18 months. We acknowledged that successful software development projects require diverse team members with different knowledge specializations to leverage each other's knowledge and produce ideas collaboratively for improving the software product (e.g., Faraj and Sproull 2000; Levina 2005; Tiwana and Mclean 2003). This involves diverse team members *collaborating effectively* by expressing or challenging, listening to, understanding, and integrating ideas in mutually beneficial ways (Levina and Vaast 2008). However, producing ideas collaboratively is particularly challenging in offshoring. In particular,

vendor employees tend to keep silent instead of contributing ideas (e.g., Jain et al. 2011; Levina and Vaast 2008; Nicholson and Sahay 2004; Winkler et al. 2008). Prior literature showed that objects such as visual designs can help diverse team members engage in practices to produce ideas (e.g., Ewenstein and Whyte 2009; Scarbrough et al. 2015). Scholars also indicated that when team members engage in practices with objects, they can change objects (e.g., Majchrzak et al. 2012). However, to date, scholars have paid less attention to the co-evolutionary dynamics between practices and boundary objects (notable exception Gal et al. 2008). In this study, we take a dynamic approach and examine how using software prototypes – that represent the current, yet provisional software end product – enable clients and vendor employees to engage in practices to produce ideas collaboratively. We also capture how idea practices and the ideas produced shape the software prototype's progression over an entire project's life. We suggest that certain idea practices may not be possible at the outset of a project when the software prototype is very basic. Yet, over time, when the software prototype progresses, more diverse idea practices may be possible that may allow clients and vendor employees to collaborate more effectively and produce different types of ideas.

1.4. Research Design

Recognizing the paucity of in-depth field studies on using software prototypes as boundary objects to enable cross-border collaboration, the research strategy in this dissertation is to closely examine data from a single case. Our findings are based on an ethnographic study that involves direct, rich and continued observations of social interactions to capture everyday practices, beliefs and lived experiences (Willis and Trondman 2000). The single case was selected because it suited the research goals to examine the key phenomena of interest exceptionally well. The knowledge asymmetries between the client and vendor were rather extreme. An entrepreneurial client from Switzerland without any software development background outsourced the development process of a novel software vision to a small team located in Vietnam. The client's vision was to provide users with a unique tool to organize, share, and view information on a common platform in various, novel ways. While the client had thought about his vision for several years, the developers were completely unfamiliar with the vision at the outset of the project. Thus, different knowledge boundaries were likely to occur over the life of the project preventing client and vendor employees from leveraging each other's knowledge specializations. Moreover, the team applied agile software development practices (Schwaber and Beedle 2002) that built heavily on

frequent interactions across borders and ensured the use of software prototypes. As part of agile practices, vendor employees continuously presented, discussed and shared as yet undeveloped and already developed designs or functionalities via screen sharing in virtual meetings with the client. Hence, it was very likely that the client and vendor engaged in different practices with the software prototype to address knowledge boundaries and leverage each other's knowledge specializations during cross-border interactions.

Data collection started in September 2013 and was completed in February 2015. We triangulated direct and indirect sources of process data. Process data comprises mainly "stories about what happened and who did what when – that is, events, activities, and choices ordered over time" (Langley 1999, p. 692). First, the primary data source was observational data. We collected observational data by attending all virtual meetings that took place 2-3 times a week between the onshore client and the offshore vendor employees. We attended these meetings as non-participant observers, recorded them, and took field notes. Second, we had access to log files and the content exchanged via the collaboration system to organize and coordinate agile software development processes. This data allowed us to access detailed information on all project documents, including requirements and changes made to them. Third, we conducted semi-structured in-depth interviews with all project participants at different points in time to capture subjective viewpoints.

Table 1 provides an overview of the three studies in this dissertation.

Table 1: Overview of Studies

Studies	*Study I:* *Transforming Software Prototypes into Boundary Objects to Bridge Knowledge Boundaries – A Revelatory Offshore-Outsourcing Case*	*Study II:* *How can Knowledge Boundaries be Effectively and Swiftly Bridged? A Longitudinal Study on Cross-Border Collaboration in Software Development*	*Study III:* *Understanding the Co-evolution of Software Prototypes and Idea Practices in Offshored Software Development*
Phenomenon to be explained	How practices help transform software prototypes into boundary objects to bridge syntactic, semantic and pragmatic knowledge boundaries	How knowledge boundaries with different levels of complexity can be permanently removed without wasting time and effort over a project's life	How software prototypes and idea practices co-evolve such that diverse team members can produce ideas collaboratively over a project's life
Data Sources	Observational data: 46 virtual meetings between client and vendor; Collaboration tool data: Log files and content exchanged	Observational data: 89 virtual meetings between client and vendor; Collaboration tool data: Log files and content exchanged, Interviews: 10	Observational data: 145 virtual meetings between client and vendor; Collaboration tool data: Log files and content exchanged; Interviews: 16
Time period	Initial 6 months	10 months	18 months
Publication Status	An earlier version was presented at and published in the Proceedings of the International Conference of Information Systems 2014 (ICIS)	A related, substantially different paper was presented at and published in the Proceedings of the European Conference of Information Systems 2015 (ECIS)	A related, substantially different paper was presented at and published in the Proceedings of the Hawaii International Conference on System Sciences 2016 (HICSS)
Contributors to Paper in this Dissertation	Winkler, Huber, Dibbern	Winkler, Huber	Winkler, Huber
Own Contribution	Major	Major	Major