Summer 6-15-2016

COLLABORATION AMIDST VOLATILITY: THE EVOLVING NATURE OF BOUNDARY OBJECTS IN AGILE SOFTWARE DEVELOPMENT

Anna Zaitsev
University of Sydney, anna.zaitsev@sydney.edu.au

Barney Tan
University of Sydney, barney.tan@sydney.edu.au

Uri Gal
University of Sydney, uri.gal@sydney.edu.au

COLLABORATION AMIDST VOLATILITY: THE EVOLVING NATURE OF BOUNDARY OBJECTS IN AGILE SOFTWARE DEVELOPMENT

Research

Anna Zaitsev, The University of Sydney, Sydney, Australia, anna.zaitsev@sydney.edu.au

Barney Tan, The University of Sydney, Sydney, Australia, barney.tan@sydney.edu.au

Uri Gal, The University of Sydney, Sydney, Australia, uri.gal@sydney.edu.au

Abstract

Agile software development has emphasis on human relations and fast paced changes in the project requirements. Extant Agile development literature focuses on team and organisational aspects of Agile development projects. The tools and artefacts used for project collaboration are currently not thoroughly researched. This research aims to find out how boundary objects are used to ensure collaboration over the Agile development project lifecycle. We selected case projects where a customer organisation was engaged with a vendor organisation in an Agile setting. Fifteen semi-structured interviews were conducted with the informants from both organisations. Our study reveals three different archetypes of boundary object: process objects, projective objects and infrastructural objects. The object archetypes respond differently to the frequency and magnitude of change. This paper enhances understanding of boundary objects aspects of Agile project development, suggests a novel way to categorise object and highlights some of the pitfalls than can result from misunderstanding object use.

Keyword: boundary object, Agile development methods, collaboration, case study

1 Introduction

Agile software development refers to a collection of different software development methods and practices based on the values of Agile Manifesto (Beck et al., 2001), which distinguishes Agile projects from more traditional, plan-driven software development approaches such as the waterfall development method (see Royce, 1970). The Agile values shift the emphasis of development from tools to humans, and from control to collaboration (Cohn and Ford, 2003). This altered emphasis can be challenging for organisations because it would require developmental practices of a vastly different nature (Conboy et al., 2011). For instance, developers using Agile methods would have to rely far more on social skills as compared to adherents of the plan-driven approaches. This is because the developers would be required to engage in a far greater extent of communication and collaboration amongst themselves, and with customers (Cockburn and Highsmith, 2001). One of the primary means of improving communications is to use prototypes and requirements descriptions to capture project-related information accurately and comprehensively (Cohn and Ford, 2003). This, in turn, necessitates a diverse set of tools and artefacts, which are used as boundary objects that facilitate the collaboration in software development projects (Winkler et al., 2014).

Boundary objects are artefacts that convey status information, discussions and agreements (Barrett and Oborn, 2010; Koskinen and Mäkinen, 2009; Yakura, 2002) across the boundaries of organisations and stakeholder groups (Star and Griesemer, 1989). They may vary in form and information capabilities, with different forms appropriate for conveying distinct types of information (Levina and Vaaste, 2005). This polymorphic nature of boundary objects makes them especially suited for supporting the diverse collaboration requirements of Agile software development projects. For instance, boundary objects can enable project stakeholders to visualize and clarify software designs (e.g. Gal et al., 2008; Winkler et al., 2014). Yet, despite their apparent usefulness in facilitating collaboration (Dybå,and Dingsøyr, 2008), the role and nature of boundary objects in the Agile software development setting has not been studied to a significant degree. There are only handfuls of articles to date that discuss the impact of boundary objects in the context of Agile development environment (e.g. Baskerville et al., 2011, Winkler et al., 2014) and their focus is not in the challenges posed by the unique nature of Agile environment.

Addressing this gap in the literature is important for two reasons. First, Agile values tend to result in a dynamic and volatile development environment (Highsmith and Cockburn, 2001), and an in-depth understanding of how different boundary objects can be used to support the various forms of collaboration that happen in this setting would increase the likelihood of project success (Carlile, 2002). Second, against the backdrop of constant change that the Agile development setting represents (Beck, 1999), boundary objects would inevitably have to be used in a fluid and flexible manner (Lee, 2007). Examining how use of boundary objects evolves over the lifecycle of an Agile project would thus enhance the ability of practitioners to manage and deploy them (as discussed by Winkler et al., 2014). Both of these reasons are all the more compelling given how Agile methods are becoming more prevalent in the contemporary software development landscape (Conboy, 2009).

To address this knowledge gap, the purpose of our study is to conduct an in-depth case study of an Agile software development project, with a particular emphasis on identifying the different types of boundary objects that are especially appropriate or prominent across the development lifecycle. In addition, we will trace how the utilisation of different types of boundary objects identified evolves over time. In doing so, we hope to provide indications to practice on how to establish and sustain effective collaboration amidst the dynamism and volatility of the Agile project setting. Corresponding to the purpose of our study, the research question that our study aims to answer is: How are boundary objects used to support collaboration in the context of Agile software development during the whole project lifecycle?

2 Literature review: Agile Software Development

The existing literature on Agile software development can be classified into three streams: practitioner guidebooks, academic research and research literature for the practitioners. The first stream consists of practitioner-oriented guidelines and *"how-to"* books that prescribe how Agile principles should be implemented in organisations (e.g. Highsmith, 2002; Schwaber, 2004). Primarily centred on the behavioural aspects of Agile implementation, this stream of literature is heavily based on the original Agile manifesto and its four core values: (1) Individuals and interactions over processes and tools, (2) working software over comprehensive documentation, (3) customer collaboration over contract negotiations, and (4) responding to change over following a plan (Beck et al., 2001).

Within this stream, some proponents of the Agile approach have developed their own variants such as Scrum (Schwaber, 2004) or eXtreme Programming (Beck, 1999, Beck, 2000), each consisting of a collection of practical guidelines. The guidelines typically highlight the different aspects of a software development project that should be accounted for with a particular emphasis on the actions of the development team. In particular, most of the variants typically assert that the project team should focus on the social aspects of the project, and seek to produce flexible, working software to enable better integration and collaboration with its customers (Highsmith and Cockburn, 2001).

The core values of the Agile manifesto (Beck et al., 2001), along with examples and guidelines of select practices that correspond to each value are summarized in Table 1.

Value	Example Practices	Implementation Guidelines
1. Individuals and interactions over processes and tools	Scrum meetings (Schwaberr, 2004) including: Daily and weekly meetings Product demonstration meetings with the extended team (Highsmith, 2002).	Meetings should preferably be face-to-face (Beck, 1999). Members of the extended team (i.e. stakeholders who do not necessarily participate in the day-to-day activities) should be involved as they may have valuable information or inputs regarding the project (Schwaber, 2004).
2. Working software over comprehensive documentation	Creation of prototypes to simulate the end products (Highsmith, 2002, Cohn, 2010)	Rapid prototyping can ensure early and constant feedback before further development (Schwaber, 2004) Prototype development should be performed simultaneously with systems development (Highsmith, 2002), in order to accommodate changes even in the late stages of development.
3. Customer collaboration over contract negotiation	Engage customers in requirements gathering and prioritisation process (Highsmith, 2002). Create stories describing requirements in lay terms (Paetsch et al., 2003, Schwaber, 2004).	Specifying requirements as stories enhances communication between the development team and the customers unfamiliar with technical jargons and conventions (Beck, 1999). Interpretation of the stories should be verified with the customers to ensure common understanding between all stakeholders (Schwaber, 2004).
4. Responding to change over following a plan	Embracing change as a source of potential improvement (Beck, 1999)	Maintain receptivity to change, which stimulates creativity that can facilitate the quicker development of the systems that are valuable for customers (Boehm, 2002).

Table 1 Summary: Agile values

The second stream of literature consists of academic studies that can be classified into four distinct categories. The first category focuses on the adoption of Agile methods (Dingsøyr et al., 2012). Studies of this category are generally seeking to identify the antecedents of adoption among members of project teams and other organisational stakeholders (e.g. Boehm, 2002). The antecedents for the adop-

tion of Agile methods that have been identified in these studies include support from the development team, customers as well as upper management (Cohn and Ford, 2003, Cao et al., 2009) and awareness of the differences with traditional methods (Boehm, 2002).

The second category centres on the management of Agile project teams and the factors that promote their effectiveness. The enabling factors that have been identified in this category of research include the level of autonomy (e.g. Sharp and Robinson, 2004) and member diversity (e.g. MacCormack et al., 2001, Nerur and Balijepally. 2007). More recent studies of this category have looked at the means of enhancing communication within globally distributed teams (e.g. Korkala and Abrahamsson, 2007; Rames et al 2006), and extended the concept of development teams to include external stakeholders such as the customers or owners of the project (e.g. Hoda et al., 2011).

The third category emphasizes team efficiency (e.g. Lee and Xia, 2010) and effective collaboration (e.g. Chow and Cao, 2008, Misra et al., 2009). Some of the most common critical success factors identified within this category of research include cultural compatibility, organisational support, customer involvement, team skills, technical skills of the team and trust. Trust, in particular, is one of the frequently cited enablers of effective collaboration (McHugh et al., 2012). For example, Ramesh et al. (2006) attributed successful communication in distributed teams to trust within the project team and between involved organisations.

The fourth category of research centres on the strategic implications and organisational value of adopting the Agile approach. Research in this category has typically examined the relationship between the adoption of Agile methods and higher-level organisational outcomes such as enterprise or system agility (e.g. Lyytinen and Rose, 2006). Other studies in this area have attempted to associate Agile principles with management philosophies such as six sigma (e.g. Qumer and Henderson-Sellers, 2008) and leanness (e.g. Conboy, 2009).

Finally, there is a stream of literature that straddles both the academic and practitioner world, describing the use of different tools and artefacts with some theoretical background. These articles delve in the particular areas of Agile development and tools that support those activities such as walls and sticky notes for planning and monitoring the progress (Sharp et al., 2009, Mateescu et al., 2015) or open source tools for testing, integration and maintenance (Azizyan et al., 2011, Nierstrasz, 2012, Dowling and McGrath, 2015). In addition to the very specific use of artefacts, few authors propose models that aim to capture the use of artefacts throughout the whole project lifecycle (Kurhmann et al., 2013, Wagenaar et al., 2015). Even thought there are few attempts to tie Agile artefacts with theory (e.g. models, comparison with Agile practices (Gill, 2015) or cognitive dimensions of Sharp et al. (2009), the stream seems to cater mainly for practitioner audiences as majority of the research in this stream lacks theoretical grounding or contributions.

Overall, the four academic categories, adoption, management, teams and organisations, are primarily centred on the behavioural aspects of Agile software development. The lack of theoretical research attention on the artefacts and tools, apart from Sharp et al., (2009), is surprising, given that the importance of tools and mediating artefacts to both systems development (e.g. Chow and Cao, 2008) and the adoption of Agile methods (e.g. Misra et al., 2009) is widely acknowledged. Accordingly, to address the lack of knowledge of the usage of boundary objects in the Agile software development setting, we turn our attention to a review of the literature on boundary objects, so as to construct a theoretical lens that serves as *"a complicated sensing device to register a complicated set of events"* (Weick, 2007, p. 16) to guide our inquiry.

3 Theoretical foundation: Boundary objects

Boundary objects are conceptual or physical artefacts that reside in the interfaces among organisations. On the one hand they are flexible enough to contain varying meanings, which arise from multiple organisations. On the other hand they are robust enough to serve as a common reference point to members of multiple organisations when they engage in mutual practice (Star and Griesemer, 1989). These

features make boundary object critical in enabling common understanding and collaboration among a diverse set of participants (Barrett and Oborn, 2010; Vlaar et al., 2008).

Research using boundary objects has discussed the multiple ways in which boundary objects can be used. For example, Lee (2007) and Barrett and Oborn (2010) describe how the use of objects impacts teamwork when the objects are used efficiently and when they are neglected. Teams, collocated or distributed, can use boundary objects as collaboration tools to define and maintain joint project requirements (Barrett and Oborn, 2010; Modi et al., 2012) or project status (Yakura, 2002) or facilitate common understanding and provide required project infrastructure (Nicolini et al., 2012).

Also focusing on cross-boundary interactions, Carlile (2002) explores the practical and political significance of boundary objects. He notes that the practical role of boundary objects is to *"establishes a shared syntax or language for individuals to represent their knowledge"* (Carlile, 2002, p. 451) and that the object *"provides a concrete means for individuals to specify and learn about their differences and dependencies"* (Carlile, 2002, p. 452). According to Carlile, the political role of objects comes from the need to facilitate a transformation of knowledge to create new knowledge.

Gal et al. (2008) take a broader perspective to examine boundary objects in the context of the identities and practices of the organisations that use them. They investigate the interdependencies between organisational identities, inter-organisational work, and the boundary objects that support this work. With a similar focus on the organisational context for their use, research has examined how boundary objects helped reconfigure boundary relations (Barrett et al., 2012) and facilitated contract negotiations between organisations, relying on their ability to create common understandings (Koskinen and Mäkinen, 2009).

Teams and organisations can use different boundary objects to convey different types of information (Levina and Vaaste, 2005). The effectiveness of the used objects is a complex issue, tied into the structure of project boundaries and to the objects themselves. Studies that look at boundary objects as communication tools, such as specifications and project management tools (Barrett and Oborn, 2010), have noted that a shift in organisational power structures can lead to unproductive use of boundary objects. The effectiveness of boundary objects is also discussed in a recent study of boundary objects in software development project by Winkler et al. (2014), who state that prototypes are effective for both bridging cross cultural boundaries as well as the three types boundaries defined by Carlile (2002): syntactic, semantic and pragmatic.

As the review above indicates, the literature on boundary objects is varied both in terms of the types and purposes of objects that have been studied, and in terms of the different contexts of their use. While some studies have examined the use boundary objects in software development projects, there is a lack of research on their use in Agile project environments.

Examining the role of boundary objects in agile settings is particularly important. Agile projects are characterized by intense, frequent, and dynamic inter-organisational interactions and communication (Cockburn and Highsmith, 2001). As such, the success of such projects heavily predicated on the ability of the participants to maintain effective collaboration in the face of such volatility (Conboy et al., 2011; Chow and Cao, 2008). However, these very characteristics of Agile projects also pose a challenge to the use of boundary objects. Because Agile projects are fast paced and encourage frequent changes to project requirements, they can challenge the capacity of project participants to use boundary objects to facilitate collaboration (Subrahmanian et al., 2003).

To address the gap in the literature on the use of boundary objects in the Agile project setting, we conducted a case study that examined how different types of objects were used by project participants in an Agile software development project.

4 Research design and methods

The case research method was used in our study for a number of reasons. First, case research is particularly appropriate for answer *"how"* questions and in our study, we are asking how boundary objects

are used to support collaboration and how does the usage evolve over time in the Agile project setting (Walsham, 2006). Second, to examine how boundary objects are used to foster collaboration between different stakeholders, an in-depth investigation to the underlying interactions between them is required (Eisenhardt, 1989). However, as Agile projects tend to unfold over an extended period of time and possibly in multiple locations, real-time data collection was deemed unpractical (Pan and Tan, 2011). Consequently, relying on the interpretation of our informants was the only feasible means of accessing the phenomenon (Klein and Myers, 1999).

The case selected for the study was an Agile software development project that involved collaboration between a customer and a vendor organisation. Both organisations were small to medium-sized businesses located in Sydney, Australia. The customer organisation is a mix of a systems development company and online retailer, while the vendor organisation is a technology-consulting firm. The aim of the project of interest was to replace a legacy system for managing customer information with an updated, more flexible and technologically more advanced system.

The case was selected as both organisations were particularly experienced in Agile development methods. In fact, the customer organisation chose the vendor on the basis of their track record and experience in Agile software development. Communication between the two organisations was conducted mainly via a customer project manager, who conveyed the requirements to the vendor, monitored the overall progress of the project, and reported on development status to internal stakeholders from the customer organisation. These internal stakeholders included the managers, internal project sponsors, as well as internal system users and testers. On the vendor side, the development team consisted of programmers, user interface designers, user experience designers and project management. Stakeholders from the customer organisation were all collocated but the development team was geographically distributed. The project thus relied heavily on communication tools and Agile meeting practices to achieve and sustain effective collaboration. This unique setting makes the case especially appropriate for the intended purpose of our study.

4.1 Data collection and analysis

Semi-structured interviews were the main source of data collection (Myers and Newman, 2007), but this was supplemented by an extensive analysis of the boundary objects used over the duration of the project. Similar approach to study boundary objects has been taken for example by Barrett and Oborn (2010). The studied boundary objects included the wireframes, visuals designs, communication tools, as well as the end product. During the first round of interviews the vendor had just completed the first version of the system that was released for the internal testing in the customer organisation. We conducted 12 interviews with seven informants from the customer organisation and three from the vendor, capturing the majority of the vendor stakeholders (refer to Table 2). Two customer managers were interviewed twice in order to first understand the project background and then for clarification. The informants represented a good mix of users, project sponsors, developers and project managers. After the interviews, the project progressed into a *"minimum viable product"* stage where the system was released to external users.

Organisation	Informants	Themes discussed
Interview Round 1, April to June 2014		
Customer	Managers A*and B*, Project Sponsors A and B, Testers A and B, Administrative User	Project events, organisation of communications, role of involved stakeholders, communication tools and mediating artefacts used, Agile practices enacted
Vendor	Manager, UX designer, Developer	Systems development process, organisation of communications, role of development team members, communication tools and mediating artefacts used, Agile practices enacted
Interview Round 2, April to May 2015		

Customer	Managers A* and C, User B	Additional communication tools, new project events since interview round 1, verification of preliminary theoretical ideas

*Table 2 The interviews *indicates an informant who was interviewed more than once*

The second round of interviews was conducted between April and May 2015, a few months after the initial release of the new system. Three additional interviews were conducted with an emphasis on clarifying the boundary objects we had identified after the first round; the new project events that had occurred since, as well as verifying our preliminary theoretical ideas (Klein and Myers, 1999).

The interview questions of both rounds were open ended and non-leading (Walsham, 1995), with a mirroring technique applied when discussing in project-specific terminology (Myers and Newman, 2007). Interview questions were adjusted based on the role of the informant, and the majority of the interviews were held at the informants' premises or nearby cafes to create a relaxed and familiar environment (Myers and Newman, 2007). One interview was held over Skype due to the location of the informant. All of the interviews were digitally recorded and transcribed (Walsham, 2006), including the interview over Skype.

Data analysis was performed in tandem with data collection (Eisenhardt, 1989). From our review of the literature on Agile software development and boundary objects, we first distilled a set of themes and subthemes that served as a theoretical lens to guide data collection (Pan and Tan, 2011). These themes were then compared with the interview data. This helped us to identify the boundary objects discussed as well as the Agile collaboration practices they were used to support. A narrative describing each major project event, transition and object usage was also created. In conjunction with the themes, the project development process and objects used in each stage of the project were visually mapped. The visual maps and narrative allowed us to trace the use and evolution of the various boundary objects across the phases of the project lifecycle (Langley, 1999). The data collected was then coded using a blend of open, axial and selective coding (Strauss and Corbin, 1990). We also established a systematic verification procedure to ensure that each finding was supported by at least two separate sources of evidence (Klein and Myers, 1999). The process of iterating between data, analysis and theory development (Eisenhardt, 1989) continued until the state of theoretical saturation was reached (Glaser and Strauss, 1967).

5 Findings

This section describes the development process, the collaboration that unfolded during the project, as well as the boundary objects that were used to support the collaboration. In order to better describe the project, we decided to divide the project into three phases: pre-project, clarification and development. This distinction helps us to relate key project event and objects to project milestones but in reality the project unravelled more organically without superimposed phases, as one would expect from an Agile project. The first *pre-project* phase marks the time before the actual engagement between the customer and the vendor began. The second and third *clarification* and *development* phases revolved around requirements elicitation and system development respectively.

The interviews we conducted touched on multiple tools and project artefacts. For the purpose of this study, we only considered those artefacts that were used to facilitate cooperation across the boundary between the customer and the vendor organisations as boundary objects. Tools and artefacts that interviewees discussed but which were only used by the customer or by the vendor internally were excluded from our analysis.

The development process, from the pre-project phase to the development phase, as well as the boundary objects used in each phase of the process, is presented in Figure 1. The figure summarises the main actions taken in each phase. It also maps the artefacts used for each action.

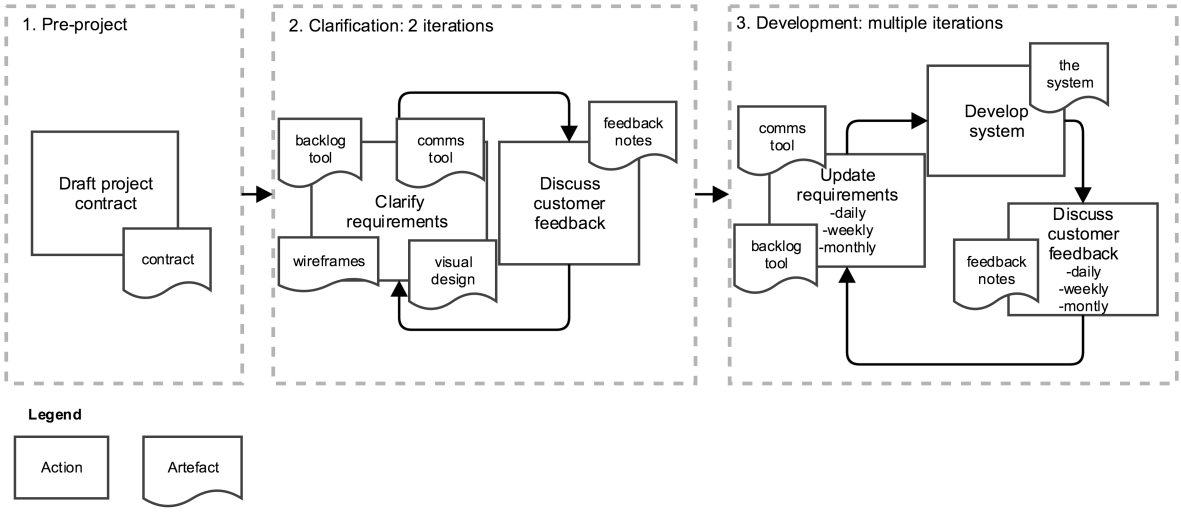

Figure 1 Project phase and boundary objects

5.1 Phase 1: Pre-project

The pre-project phase began when the customer finalized their internal vendor selection process and engaged the vendor. The engagement started with the crafting of the contract between the vendor and the customer. During the project, Scrum practices (Schwaber, 2004) were used between the customer and the vendor. Traditional approaches, such as the waterfall method, were not considered for this project and stakeholders from the customer organisation were adamant on utilizing Agile. A project sponsor explained:

"We didn't want to sit there, waiting for nine months for them to deliver an end product, we wanted something now" – Customer Project Sponsor A

The contract of this project, which is in essence a boundary object, was designed to accommodate the flexibility inherent in Agile processes. Defined collaboratively by the customer and the vendor, the scope of the project was kept deliberately vague. While the contract outlined the schedule and budget, it allowed the customer to terminate the project at any time, as one of the vendor's managers explained:

"We like to say that the client can walk away anytime… When they walk away they've got the software delivered because we are delivering on day-to-day basis..." –Vendor Manager

The pre-project phase ended when both parties signed the contract. The second phase focused on clarifying and refining the requirements of the project, which were very loosely described in the contract of the initial phase.

5.2 Phase 2: Clarification

After the contract between the vendor and customer was defined, the development team began gathering, consolidating, and clarifying the requirements for the system design and functionalities.

The main collaborative tools during the second phase were wireframe models, a simplified prototype representation of the end system. The wireframes were used to communicate and refine system requirements. The wireframes used in the project were relatively sophisticated. They captured the main features and functionalities of the system and presented them in the form of an interactive prototype.

The wireframes were accompanied by a visual design of the system, outlining the fonts, colours and other visual elements. The feedback given by customer stakeholders was captured and used for im-

provements to the wireframes and the visual designs. Members of both the customer and vendor organisations described this process:

"Our UX designer would create a number of interactive wireframes that she presented to the business and they could click through and play with the wireframe to get a feel of what we had in mind" – Vendor Manager

In concurrence with the wireframes, the project team also set up other collaboration tools. The two main collaboration tools were the backlog tool and communications tool. Both tools were used to convey information from the vendor to the customer and vice versa. The backlog tool was used to capture and consolidate the requirements collected from the customer notes and from the wireframes. The communications tool was used to discuss and clarify the requirements in the backlog tool. A manager from the customer organisation commented of the relations of the backlog tool and the communication tool:

"We communicate normally via an online chat tool. The way that we communicate is trough threads and we find it is fantastic for what we do... In fact it is more efficient than getting updates in meetings and all that sort of thing…[The backlog tool] is for our backlog management and our general priorities management as well. But we do put communication into [communication tool] because it is just easier to reference to a feature than within [backlog tool]. Just for example, a feature has been developed to a certain point and then you can ask people, "ok, where is it up to"... You want to keep communication, very specific feature communication, to [backlog tool]. The [communication tool] it's more about conceptual conversations. –Customer Manager A

The customer and vendor went through two main iterations that entailed major wireframe and requirements updates. After these two rounds of changes, the wireframes and documented requirements were deemed to be clear enough by both parties. This point marked the beginning of the project's third phase – the development phase.

5.3 Phase 3: Development

Like the clarification phase, the development phase also unfolded in an iterative manner. The development team had short daily meetings to discuss the development and unit testing work it had accomplished as well as upcoming work. By the end of each week, the team released a functional new version of the system. The customer then reviewed this version and provided feedback to the vendor using feedback notes and a testing spreadsheet. The requirements were then updated based on the customer feedback. The monthly iterations consisted of reorganising the larger contours of the project such as the priorities of wider scope requirements according to the customer feedback.

"In July we will have beta version, which will have the most simplified version of the software. Not all the bells and whistles but most the key features and things that a program needs in order to operate. The extra features are scheduled down the track for implementation." –Customer Project Sponsor A

During the development phase, the functioning system itself replaced the wireframes as the main communicative point of reference and boundary object for the two organisations. Because the system was updated and released weekly, it captured customer feedback more faithfully and accurately than previous versions of the system, as well as the wireframes, which became obsolete because they were not being updated anymore. The visual design, which was deemed sufficiently clear by the end of the second phase, was simply merged with the actual system and thus became part of the weekly update cycle.

"So it went from the wireframes to a sort of mock up. Then the guys were just building features and then from there, as soon as there was something to show us, we could dib in and dib out and test and have a look and see what is being developed." –Customer Project Sponsor B

Additional tools were used to facilitate communication within and across the two organisations. The communication between vendor and customer stakeholders was logged in the testing spreadsheet. Emerging ideas, improvements, and technical specifications continued to bring about further changes

and new system requirements. The requirements updates were stored in the requirements backlog tool, which also contained the status of existing and future features.

By the end of the second round of interviews, the development had progressed beyond the internal testing. The product had reached the stage of a "minimum viable product", the version that the customer considered to be complete enough for external release and had just been released to the first set of the customer's external clients.

6 Discussion

Based on our interview data, we identified three types of boundary object archetypes. We traced the evolution of three archetypes and discovered varying levels of object dynamism across the three project phases. Moreover, our data shows that there is a distinct hierarchy between the object archetypes. This section will discuss the archetypes of boundary objects identified, the varying levels of dynamism of each archetype across the different project phases, as well as the hierarchy between them.

6.1 Infrastructural boundary objects

Infrastructural boundary objects constitute the embedded framework that enabled and guided processes, systems and tools throughout the project (Star and Ruhleder, 1996). The infrastructural objects are crafted either at the very beginning of the project or before project participants begin to establish shared understanding of what the project will encompass (Koskinen and Mäkinen, 2009). Infrastructure *"occurs when tension between local and global is resolved"* (Star and Ruhleder, 1996. p. 114). In our research case, the tension was resolved when the project foundations are set up to enable the existence of the other object types.

The project contract, identified as an infrastructural object in the case project, provided legitimacy for the project. The contract was used to provide common understanding of the project premises across the boundary between the organisations. Infrastructural objects cannot be replaced by any other means of communication. When these objects are used in Agile environment, the changes that occur in the project requirements should have minimum impact on the functionality of these objects as Agile development methods require very flexible infrastructure objects. More rigid objects that would require constant adjustments could potentially hinder the Agile development process with overhead activities.

6.2 Process boundary object

Process boundary objects refer to all the different tools used for the purpose of documentation. Containing different types of textual and visual information, these objects convey both status information of the requirements and customer feedback on the requirements, as well as status information of the entire project. The process objects supporting Agile processes in our case study include backlogs, communication tools and feedback notes.

Process objects are used for documenting features and feature changes. A trait of process objects is that they can be potentially replaced with other means of communication such as meetings, had the stakeholders been all located in the same premises. However, the importance of process objects increases if the development team is not collocated as was the case in the project we studied, and which is a situation in the majority of contemporary software projects (Šmite et al. 2010). Geographical distribution hinders the use of other communication means, and increases the importance of having effective process objects (Turner et al. 2006).

6.3 Projective boundary object

Projective boundary objects are those that provide a preview of the project deliverables. These objects are used to convey the desired outcomes of the project, what the system will look like eventually relative to its current state. These objects include different visual representations of the developed system,

such as, wireframes, visual design documentation, and the actual functional system (i.e. the work-in-progress).In the case project the wireframes and visual designs and the early work-in-progress system were used as a prototype that facilitated the discussion of the project goals. Prototypes are one of the more often discussed topics in boundary object literature (for example Gal et al., 2008; Winkler et al., 2014) but the projective objects category extends beyond only prototypes, including a wider range of visual objects that can be used to negotiate common goals across boundaries.

Projective boundary objects provide material for feedback that is captured in process boundary objects. The objects of this archetype tend to be malleable in that requirement changes can be reflected in these objects. Desired outcomes are iteratively refined and clarified to evolve into functional versions of the system, and this evolution continues until the end of the project lifecycle.

Table 2 provides a summary of boundary object archetypes, their characteristics, as well as corroborating evidence from our case data.

Boundary object	Characteristics	Corroborating Evidence
Infrastructural boundary objects: Contract	To facilitate and provide a baseline for other boundary objects Must support the chosen development methodology Star and Ruhleder (1996)	*"We actually wrote up a contract that defined that {the vendor} will be using Agile methodologies such as weekly stand ups. The are able to change scope and we would work in a time-and-materials way…It was very loosely written." – Customer Manager B*
Process boundary objects: Backlog tool Communication tool Feedback notes	To store documents, feedback and discussion data, and project status information. Linked with other Process boundary objects (e.g. complementary data, updates)	*"Quite often if I look up something in [backlog tool] there will be some functionality but it does not have enough information, so its just quicker to ask directly in [communication tool][from customers]" –Vendor Developer*
Projective boundary objects: Wireframes Visual design The system	To provide a preview of project deliverables including functionality and visual look-and-feel. Allow input for customer feedback and new requirement ideas. Winkler et al. (2012)	*"She used this [software] that just brought [the wireframes] to life and made us feel "this is what is going to be like". – Customer Project Sponsor A*

Table 2 Boundary object archetypes

6.4 Object dynamism and hierarchy

Our data analysis indicated that throughout the project, the three archetypes of boundary objects presented different levels of dynamism. Namely, they underwent changes that were different in terms of their frequency and magnitude. Frequency refers to the number of times change occurs in an object's state during a time period. Magnitude refers to the perceived significance of the object's change from its previous form. Slater and Narver (1994) use the term "dynamism" in similar manner. As this study was examining the collaboration across the boundary between the customer and the vendor, the dynamism presenter here is observed from the perspective of the customer stakeholders. The vendors might have a different view on the dynamism due to their different project activities but their perspective is not explored further this study.

Figure 2. illustrates the dynamism of the three object archetypes during the three project phases. The different dots represent explanatory instances of object use over time as seen from the customer perspective. The higher on the y-axis the illustrative dots are, the more significant are the changes that have occurred during these instances of collaboration and communication. For example, during the

clarification phase, the projective objects undergo more drastic changes than during the development phase of the project. The customers see first rapidly changing versions (e.g. the wireframe prototype) of the projective objects that is later modified only periodically, during the releases of new versions.

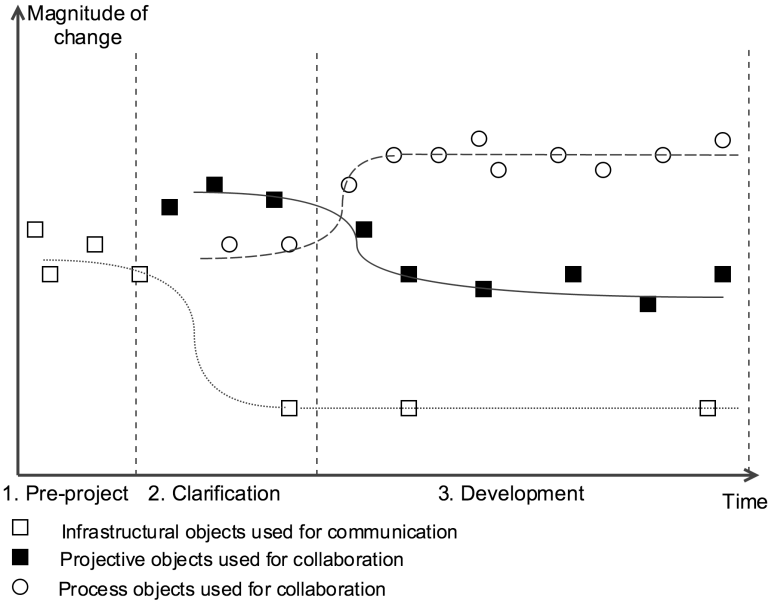

Figure 2 Object dynamism

Infrastructural objects were introduced in the pre-project stage. As the project progressed into the clarification and development phases, the infrastructural objects stabilized and receded into the background when the conflict, discussed by Star and Ruhleder (1996), was resolved. After the initial negotiations there were minimal changes to the contract. The contract was described to be flexible and transparent:

"We have a certain amounts of budget, which brings it to a certain timeframe with them, and the vendor is fully aware of what that timeframe is. Now wether we get sign off or more budget in order to continue on with the projects, there's something that might happen in the future." Customer project manager

Projective object, introduced in the clarification phase, were used to establish a common understanding of the project's end goals. The objects underwent several rapid and significant changes to accommodate ongoing shifts in requirements. After the initial clarification, the changes the projective objects underwent in the development phase were less frequent and significant. This was due to the shift from earlier prototypes such as the wireframes types into more complete prototype such as the work-in-process version of the system, which decreased the object dynamism. The main reason for the decrease was that at the development stage, the changes in the system that the customer organisation observed were scheduled and controlled in order to enable the customer feedback.

"I was the first tester of what they thought was a first good version of the system. <The project team> wanted someone's input throughout the process that they were going thought and making sure that they were on the right tracks. ... When the minimum viable product version was released it was more user friendly, it was further developed form the initial stages." –Customer tester A

Process objects were also introduced in the clarification phase, but unlike projective objects, the dynamism of their change increased as the project progressed into the development phase. The process objects were used for daily communication, which became more important in the development phase

as the development team engaged with additional stakeholders, such as the internal testers. Gradual changes in projective objects, as well as the new feedback from stakeholders, needed to be captured in the process objects. This meant that the project team had to update all process objects so that they contain updated information and its history for cross-referencing. Vendor manager described the constant update process of the requirements documentation in the backlog tool:

"If [the customer] finds a bug, he will create a bug ticket in [backlog tool]. A developer will work thought that bug and the customer will get notified when it will be ready for acceptance testing…And he also involves his testing team people. It is not just bugs but also feature changes that are added and updated." – Vendor Manager

The varying patterns of their dynamism and their distinct features suggest that the three archetypes of boundary objects are hierarchically ordered such that one archetype enables the functioning of the others. Firstly, the infrastructural objects provide the framework for the definition, functioning, and meaning of processes and objects.

Secondly, the projective objects contain a vision of the final product that is being developed. Wireframes and visual designs offer a tangible (albeit dynamic) representation of the system and serve as a reference point for discussions around how to achieve it. The system itself, in its varying stages of development, offers another, more concrete, visualisation of the project's end goal. On the one hand, these objects only make sense within a procedural infrastructure. On the other hand, the vision that they espouse informs the definition of the system requirements and the processes to realise them, as captured in the process objects.

Thus, a third connection exists between the projective objects and the process objects. The process objects reflect the information contained in the projective objects. However, while projective objects provide a vision of the finished system, the information in the process objects translates this vision into a more practical language (i.e., status information, discussion data) that provides a more in-depth perspective of the mechanics of how to get there.

7 Concluding remarks

By addressing the research question set forth at the beginning of this paper, this study makes a number of important theoretical contributions. First, this study is one of the earliest that examines the role and evolution of boundary objects in the context of Agile software development projects. In doing so, it complements the existing studies that focus primarily on the behavioural aspects of Agile software development (e.g. Chow and Cao, 2008, Misra et al., 2009) and contributes towards the understanding of the material aspects of the phenomenon (Cecez-Kecmanovic et al., 2014).

Second, this study has identified three boundary objects archetypes that are salient to Agile software development, provided indications on their varying levels of dynamism across the phases of the development lifecycle, as well as the hierarchy between them. All of these are conceptual innovations that can be used as the basis of developing a more comprehensive typology of boundary objects, or a more nuanced and context-specific theory surrounding their use.

In terms of contributions to practice, practitioners could use our findings to better understand the functions of the different boundary objects archetypes, and how each of them may be leveraged to enhance collaboration. Our study highlights a number of potential pitfalls that Agile practitioners should be wary of. For example, if infrastructural boundary objects such as contracts are too rigid, they can hinder change and thus have a negative impact on the other objects. Moreover, process boundary objects that change frequently could be vulnerable to boundary object erosion (Subrahmanian et al., 2003). Keeping process boundary objects up to date might help to avoid project status confusion, or disparities in requirement descriptions. This will aid the maintenance of the efficiency of the boundary objects (Carlile et al., 1997). All in all, it is hoped that Agile practitioners can use the findings of our study as a reference to tailor their approach to using boundary objects while avoiding the potential pitfalls, so that they can better harness their potential for facilitating collaboration.

References

Azizyan, G., M.K. Magarian and M. Kajko-Matsson (2011). "Survey of Agile Tool Usage and Needs," *Agile Conference (AGILE), 2011*: IEEE, 29-38.

Barrett, M. and E. Oborn (2010). "Boundary object use in cross-cultural software development teams." *Human Relations* 63 (8), 1199-1221.

Barrett, M., E. Oborn, W.J. Orlikowski and J. Yates (2012). "Reconfiguring boundary relations: Robotic innovations in pharmacy work." *Organization Science* 23 (5), 1448-1466.

Baskerville, R., J. Pries-Heje and S. Madsen (2011). "Post-agility: What follows a decade of agility?" *Information and Software Technology,* 53 (5), 543-555.

Beck, K. (1999). "Embracing change with extreme programming." *Computer,* 32 (10), 70-77.

Beck, K. (2000). *Extreme programming explained: embrace change,* Addison-Wesley Professional.

Beck, K., M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J, J. Highsmith, A. Hunt and R. Jeffries, (2001). "*The agile manifesto*." URL: www.agilemanifesto.org

Boehm, B. (2002). "Get ready for agile methods, with care". *Computer,* 35 (1), 64-69.

Cao, L., K. Mohan, P. Xu and B. Ramesh (2009). "A framework for adapting agile development methodologies." *European Journal of Information Systems,* 18 (4), 332-343.

Carlile, P. R. (2002). "A pragmatic view of knowledge and boundaries: Boundary objects in new product development." *Organization science,* 13 (4), 442-455.

Carlile, P. R., J.E. Dutton and M.S. Feldman (1997). *Understanding knowledge transformation in product development: Making knowledge manifest through boundary objects.* University of Michigan.

Cecez-Kecmanovic, D., R. D. Galliers, O. Henfridsson, S. Newell and R. Vidgen (2014). "The sociomateriality of information systems: current status, future directions." *MIS Quarterly,* 38 (3), 809-830.

Chow, T. and D.-B. Cao (2008). "A survey study of critical success factors in agile software projects." *Journal of Systems and Software,* 81 (6), 961-971.

Cockburn, A. and J. Highsmith (2001). "Agile software development: The people factor." *Computer,* 34 (11), 131-133.

Cohn, M. (2010). *Succeeding with agile: software development using Scrum.* Pearson Education.

Cohn, M. and D. Ford (2003). "Introducing an agile process to an organization." *Computer,* 36 (6), 74-78.

Conboy, K. (2009). "Agility from first principles: reconstructing the concept of agility in information systems development." *Information Systems Research,* 20 (3), 329-354.

Conboy, K., S. Coyle, S, X. Wang and M. Pikkarainen (2011).. "People over process: key people challenges in agile development." *IEEE Software*, 28 (4), 48-57

Dingsøyr, T., S. Nerur, V. Balijepally and N.B. Moe (2012). "A decade of agile methodologies: Towards explaining agile software development." *Journal of Systems and Software,* 85 (6), 1213-1221.

Dowling, P. and K. McGrath (2015). "Using free and open source tools to manage software quality." *Communications of the ACM*, 58 (7), 51-55.

Dybå, T. and T. Dingsøyr (2008). "Empirical studies of agile software development: A systematic review." *Information and software technology,* 50 (9), 833-859.

Eisenhardt, K. M. (1989). "Building theories from case study research." *Academy of management review,* 14 (4), 532-550.

Gal, U., K. Lyytinen and Y. Yoo (2008). "The dynamics of IT boundary objects, information infrastructures, and organisational identities: the introduction of 3D modelling technologies into the architecture, engineering, and construction industry." *European Journal of Information Systems,* 17 (3), 290-304.

Glaser, B. G. and A. L. Strauss (1967). *The discovery of grounded theory: Strategies for qualitative research,* Aldine Pub. Co.

Gill, A. Q. (2015). "Distributed Agile Development: Applying a Coverage Analysis Approach to the Evaluation of a Communication Technology Assessment Tool," *International Journal of e-Collaboration (IJeC)* 1(11), 57-76.

Highsmith, J. (2002) *Agile software development ecosystems,* Addison-Wesley Longman Publishing Co., Inc.

Highsmith, J. and A. Cockburn (2001). "Agile software development: The business of innovation." *Computer,* 34 (9), 120-127.

Hoda, R., J. Noble and S. Marshall (2011). "The impact of inadequate customer collaboration on self-organizing Agile teams." *Information and Software Technology, 53*(5), 521-534.

Klein, H. K. and M.D. Myers (1999). "A set of principles for conducting and evaluating interpretive field studies in information systems." *MIS quarterly*, 23 (1), 67-94.

Korkala, M. and P. Abrahamsson (2007). "Communication in distributed agile development: A case study." In *Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on*: IEEE. 203-210.

Koskinen, K. U. and S. Mäkinen (2009). "Role of boundary objects in negotiations of project contracts." *International Journal of Project Management,* 27 (1), pp. 31-38.

Kuhrmann, M., D. Mendez Fernandez and M. Grober, (2013). Towards artifact models as process interfaces in distributed software projects. In *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on* IEEE, 11-20.

Langley, A. (1999). "Strategies for theorizing from process data." *Academy of Management review,* 24 (4), 691-710.

Lee, C. P. (2007). "Boundary negotiating artifacts: Unbinding the routine of boundary objects and embracing chaos in collaborative work." *Computer Supported Cooperative Work (CSCW),* 16 (3), 307-339.

Lee, G. and Xia, W. (2010). "Toward agile: an integrated analysis of quantitative and qualitative field data." *Management Information Systems Quarterly,* 34 (1), 87-114

Levina, N. and E. Vaast (2005). "The emergence of boundary spanning competence in practice: Implications for implementation and use of information systems." *MIS Quarterly*, 29 (2), 335-363.

Lyytinen, K. and G. M. Rose (2006). "Information system development agility as organizational learning." *European Journal of Information Systems,* 15 (2), 183-199.

MacCormack, A., R. Verganti and M. Iansiti (2001). "Developing products on "Internet time": The anatomy of a flexible development process." Management science, 47 (1), 133-150.

Mateescu, M., M. Kropp, R. Burkhard, C. Zahn and D. Vischi (2015).. aWall: A Socio-Cognitive Tool for Agile Team Collaboration using Large Multi-Touch Wall Systems. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*, ACM. 361-366

McHugh, O., K. Conboy and M. Lang (2012). "Agile practices: The impact on trust in software project teams." Software, IEEE, 29 (3), 71-76.

Misra, S. C., V. Kumar and U. Kumar (2009). "Identifying some important success factors in adopting agile software development practices." *Journal of Systems and Software,* 82 (11), 1869-1890.

Modi, S., P. Abbott and S. Counsell (2013). "Negotiating common ground in distributed agile development: A case study perspective." In *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on*: IEEE. 80-89.

Myers, M. D. and M. Newman (2007). "The qualitative interview in IS research: Examining the craft." *Information and organization,* 17 (1), 2-26.

Nerur, S. and V. Balijepally (2007). "Theoretical reflections on agile development methodologies." *Communications of the ACM,* 50(3), 79-83.

Nierstrasz, O. (2012). Agile software assessment with Moose. *ACM SIGSOFT Software Engineering Notes*, 37(3), 1-5.

Nicolini, D., J. Mengis,and J Swan (2012). "Understanding the role of objects in cross-disciplinary collaboration." *Organization Science*, 23 (3), 612-629.

Paetsch, F., A. Eberlein, and F. Maurer (2003). "Requirements engineering and agile software development." In *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*: IEEE Computer Society. 308-308.

Pan, S. L. and B. Tan,(2011). "Demystifying case research: A structured–pragmatic–situational (SPS) approach to conducting case studies." *Information and Organization,* 21(3), 161-176.

Qumer, A. and B. Henderson-Sellers (2008). A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software,* 81 (11), 1899-1919.

Ramesh, B., L. Cao, K. Mohan and P. Xu (2006). "Can distributed software development be agile?" *Communications of the ACM,* 49 (10), 41-46.

Royce, W. W. (1970). "Managing the development of large software systems." In *proceedings of IEEE WESCON*: Los Angeles.

Schwaber, K. (2004). *Agile project management with Scrum,* Microsoft Press.

Sharp, H. and H. Robinson (2004). "An ethnographic study of XP practice." *Empirical Software Engineering,* 9 (4), 353-375.

Sharp, H., H. Robinson, and Petre, M. (2009).. The role of physical artefacts in agile software development: Two complementary perspectives. *Interacting with computers*, 21(1-2), 108-116.

Slater, S. F. and J. C. Narver (1994). "Does Competitive Environment Moderate the Market Orientation-Performance Relationship?".*Journal of Marketing*, 58 (1), 46–55

Šmite, D.,, C. Wohlin,, T.Gorschek and R. Feldt, (2010). "Empirical evidence in global software engineering: a systematic review." *Empirical software engineering,* 15(1), 91-118.

Star, S. L. and J. R. Griesemer (1989). "Institutional ecology,translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39." *Social studies of science,* 19 (3), 387-420.

Star, S. L. and K. Ruhleder (1996). "Steps toward an ecology of infrastructure: Design and access for large information spaces." *Information systems research,* 7(1), 111-134.

Strauss, A. and J. M. Corbin, (1990). *Basics of qualitative research: Grounded theory procedures and techniques,* Sage Publications, Inc.

Subrahmanian, E., I. Monarch, S. Konda, H. Granger, R. Milliken and A. Westerberg (2003). "Boundary objects and prototypes at the interfaces of engineering design." *Computer Supported Cooperative Work (CSCW),* 12(2), 185-203.

Turner, W., G. Bowker, L. Gasser and M Zacklad (2006). Information infrastructures for distributed collective practices. *Computer Supported Cooperative Work (CSCW)*, 15 (2), 93-110.

Vlaar, P. W., P.C. van Fenema and V. Tiwari (2008). "Cocreating understanding and value in distributed work: How members of onsite and offshore vendor teams give, make, demand, and break sense." *MIS quarterly,* 32 (2), 227-255.

,am, G. (1995). "Interpretive case studies in IS research: nature and method." *European Journal of information systems,* 4 (2), 74-81.

Walsham, G. (2006). "Doing interpretive research." *European journal of information systems,* 15 (3), 320-330.

Wagenaar, G., Helms, R., Damian, D., & Brinkkemper, S. (2015). Artefacts in Agile Software Development. In *Product-Focused Software Process Improvement*, Springer International Publishing. 122-148

Weick, K. E. (2007). "The generative properties of richness." *Academy of management journal,* 50 (1), 14-19.

Winkler, M., T. Huber and J. Dibbern (2014). The Software Prototype as Digital Boundary Object–A Revelatory Longitudinal Innovation Case. In *proceedings of International Conference on Information Systems ICIS 2014, Auckland, New Zealand, 2014*

Yakura, E. K. (2002). "Charting time: Timelines as temporal boundary objects." *Academy of Management Journal,* 45 (5), 956-970.