Too Tight - Too Loose: Controlling Lessons from a Failed IS Development Project

Subasinghage Maduka Nuwangi

Business Information Systems Department
Auckland University of Technology
Auckland, New Zealand
Email: maduka.subasinghage@aut.ac.nz

Darshana Sedera

Faculty of Information Technology
Monash University
Melbourne, Australia
Email: darshana.sedera@gmail.com

Abstract

Information Systems Development outsourcing (ISD-outsourcing) projects are notorious for not providing agreed deliverables within the stipulated time and budget. More alarmingly, it is estimated that 19% of ISD-outsourcing projects fail outright (Hastie and Wojewod 2015). This study investigates an ISD-outsourcing failure case which was attributed to issues in the requirements engineering process. Adopting a 'control theory' perspective and leveraging case study approach, we examine how the weaknesses in control portfolio contributed to project failure. We attribute the failure not to the absence of control portfolios, rather (i) employing particular control mechanisms incorrectly depending on the phase of requirements engineering process, (ii) dominance of a particular control mechanism, and (iii) insufficiency of particular control mechanisms.

Keywords

Information Systems Development Outsourcing, Control Theory, Requirement Engineering

1   Introduction

According to Gartner (2017), the worldwide IT outsourcing services market reached about 283.5 billion U.S. dollars in 2016. A survey conducted with industry professionals[1] indicated that 72% of their IT functions have been outsourced and there will be a further 31% of increase in the future (Deloitte 2016). Information Systems Development outsourcing (ISD-outsourcing) is one of the most popular type of outsourcing, with strong and continuous growth in ISD-outsourcing initiatives (Gregory et al. 2013). ISD-outsourcing is a contract-based relationship between client and vendor organization wherein the client contracts out all or part of its ISD activities to the vendor (Khan et al. 2011). These projects require continuous engagement between the client and the ISD vendor to exchange ideas, information and efforts.

ISD-outsourcing projects commence with a contractual agreement between the client and the ISD vendor. These projects consist of several stages such as requirements engineering, design, development, testing and implementation (Boehm 1988). The requirements engineering is particularly important step in ISD-outsourcing projects, where the focus is on understanding the development requirements (Pohl 1996). Projects often fail due to mis-specified requirements (Mishra et al. 2008), conflicting requirements (Salado and Nilchiani 2016) and in general for lacking a clear understanding of the requirements (Sim and Brouse 2014). According to Pohl (1996, p. 3), requirements engineering is "a systematic process of developing requirements through an iterative co-operative process of analysing the problem, documenting the resulting observations in a variety of representation formats, and checking the accuracy of the understanding gained." Nuseibeh and Easterbrook (2000) discussed the stages of requirements engineering process as; 1) requirements elicitation, 2) requirements analysis and modelling, 3) requirement communication, 4) agreeing with the requirements, and 5) evolving requirements. Requirements engineering process entails iterative discussions between the client and the team members of ISD vendor such as consultants, project managers and technical staff. It is also a vital process that identifies the budgetary constraints and technical limitations (Saiedian and Dale 2000). Requirements engineering is conducted in multiple iterations, where the requirements are gathered, updated, verified, confirmed and documented. Once the requirement documents are developed, they are passed on to the technical staff to develop the system with agreed specifications.

Requirements engineering process is inherently complex. First, it entails multiple stakeholders who rarely agree on common value propositions (Nuseibeh and Easterbrook 2000). Second, a third-party organization discovering the business requirements from the client organization is always open for misinterpretation and mis-specification (Mishra et al. 2008; Sedera et al. 2014). Third, the conversion of business requirements into technical requirements often lead to misunderstandings (Tiwana 2004). In addition, complexities could arise through lack of appropriate skills (Tiwari et al. 2012), inadequate knowledge transfer mechanisms (Pee et al. 2010) and over promising of requirements by the ISD organization (Ehrhart 2002; Nuwangi and Sedera 2017). Given the complexity of requirements engineering process in ISD-outsourcing projects, the selection of appropriate control mechanisms for each stage of requirements engineering process is crucial for project success.

ISD-outsourcing projects are notorious for not providing the agreed deliverables on time (Nakatsu and Iacovou 2009; Savolainen et al. 2012; Srivastava and Teo 2012). As per Standish Group (2014) average only 16.2% of ISD-outsourcing projects are completed within the expected time and budget. More alarmingly, it is estimated that 31.1% of ISD projects are cancelled before completion (Standish Group 2014) and 19% of projects completely fail (Hastie and Wojewod 2015). However, rarely we receive an opportunity to investigate the failed projects (Chua and Lam 2005). It is said that failure cases can provide insights that are often ignored in investigations of successful projects (Lyytinen and Robey 1999).

This study investigates an ISD-outsourcing failure case which was attributed to the inefficiencies and issues of the requirements engineering process. The present study addresses the following research question: "how the weaknesses in the control portfolio contribute to project failure?" This posthumous view provides a rare insight into where and how ISD-outsourcing firms should focus on formal and informal controls. The analysis and recommendations presented here provide insight into the landscape of control mechanisms in ISD-outsourcing. The paper proceeds in the following manner. Next section discusses the theoretical background of the research. Then, it explains the case and research

---

[1] More than 85% of the respondents were from companies with over $1 billion annual revenue.

methodology followed by preliminary results section. The paper concludes with a discussion, which includes summary of results and research implications.

2 Theoretical Background

Requirements engineering process consists of five main stages; 1) requirements elicitation, 2) requirements analysis and modelling, 3) requirement communication, 4) agreeing with the requirements, and 5) evolving requirements (Nuseibeh and Easterbrook 2000). Requirement elicitation, the first stage of requirements engineering process is focused on capturing client requirements, system boundaries and goals. Interviews, observations, focus groups and brainstorming can be considered as techniques that can be used to capture client requirements during requirement elicitation (Paetsch et al. 2003). Requirement analysis and modelling is where the collected requirements are analysed for consistency, completeness and feasibility (Paetsch et al. 2003). Requirements modelling techniques such as data-flow models and object-oriented approaches can be used to depict the requirements. Requirement documentation is important method of requirement communication between multiple stakeholders. Goal of requirement documentation is to ensure that requirements can be read, analysed and validated. Since it is difficult to identify all requirements initially, the development of requirement documents can be considered as an incremental process where the requirements are added to the documents successively (Mishra et al. 2008). Requirements are validated by certifying that the requirements are an accurate representation of Information System to be implemented. After the requirement validation, the stakeholders agree and sign-off the requirements. Agreeing upon requirements can be difficult as the stakeholders' may have diverse goals. Since ISD-outsourcing projects consist of volatile client requirements (Gefen et al. 2008), managing requirement changes is an important step of requirements engineering process.

2.1 Control Theory

Control Theory is one of the most prominent theoretical foundations to study ISD-outsourcing projects (Wiener et al. 2016). There are four types of management control modes that are commonly considered in control theory literature that can be used in assisting complex project management. Formal controls involve managing employees through mechanisms such as performance evaluations in which either the outcomes or behaviours of the employees are measured, evaluated and rewarded (Kirsch 1996). Formal controls are further subdivided into outcome-based and behaviour-based control modes, where the outcome-based modes specify the expected outcomes of projects, and the behaviour-based modes influence the behaviours (Kirsch 1996). Common project management techniques companies employ in ISD-outsourcing projects, like the requirement documents, project plans and regular meetings, fall under the outcome and behaviour controls of the control theory (Choudhury and Sabherwal 2003; Nuwangi et al. 2014). The informal controls like social and people strategies are also commonly employed in ISD-outsourcing projects. The informal control consists of clan and self-controls. Ouchi (1978) explains clan control as promoting common values and beliefs within a clan, which is defined as a group of individuals who share a set of common goals. Unwritten clan-control mechanisms can promote a sense of unity and a shared objective in disparate projects. Self-control occurs when the employees of the company control their own actions (Kirsch 1996). The formal ones – outcome and behavioural control modes are typically suitable for well-structured projects, while the informal – clan and self-controls are suitable for situations with evolving structures (Kirsch 1996).

As per control theory, ISD-outsourcing projects can be managed by implementing a portfolio of controls, which consists a mix of formal and informal controls (Choudhury and Sabherwal 2003; Kirsch 1996). Previous research discussed the use of control portfolios to manage ISD-outsourcing projects (Choudhury and Sabherwal 2003; Nuwangi 2016; Wiener et al. 2016), however to the best of the authors' knowledge, no study has specifically focused on identifying appropriate control mechanisms as per the stages of requirements engineering process. Requirements engineering is a complex and unpredictable process which combines ideas, perspectives and objectives of multiple stakeholders (Abrahamsson et al. 2017; Pandey et al. 2010). Since requirements engineering process provides the base for all other project stages such as software coding and testing, it is important to select appropriate control mechanisms as per the stages of requirements engineering process. Selecting appropriate control mechanisms as per the requirement engineering process enables the project team to constantly act upon evolving requirements (Abrahamsson et al. 2017). An effective control portfolio has the potential to deploy a combination of formal and informal controls based on the situation. Much of the existing knowledge on control portfolio suggests deriving a portfolio 'based on the context' (Wiener et al. 2016). While this is true for most situations, prescriptive guidelines on which composition of controls work better, and which control compositions are worst at 'known' situations are absent in the literature. A well-developed

prescriptive guidelines of a control portfolio at the requirements engineering would: 1) ensure that ISD vendor completes agreed task on-time with agreed service level agreements (Kirsch 1996), 2) ensure cooperation amongst individuals (Choudhury and Sabherwal 2003), 3) monitor, evaluate and provide feedback, especially in issues (Kirsch 1996), and overall 4) increase team performance (Wiener et al. 2016).

3   Introducing the Case and Methodology

As the phenomenon being examined is relatively new and unexplored, we follow the case study method to capture the richness of the context in which the phenomenon is situated (Yin 2009). Three conditions formed the benchmark for the selection of the research site. First, the vendor should have a successful track record of managing ISD projects. Second, the vendor must execute multiple ISD-outsourcing projects at the time of our interviews and third, the vendor must have been involved in ISD-outsourcing for at least five years, to have mature and stable controlling structures. Vendor-StockEx[2], was a large ISD-outsourcing company engaging in stock exchange related software application development. Vendor-StockEx has been in the business for over 10 years, employing over 300 staff. Aforementioned characteristics convinced the section of Vendor-StockEx as our target case. Vendor-StockEx developed capital market automation software including the functionalities of multiple trading methods, asset classes and market structures.

3.1   The Failed Project

The purpose of this project was to develop a post-trade application, which provides clearing and settlement of trades after execution. The solution developed in this project was characterized by complex trade processing methods, which were highly integrated with clearing and settlement procedures. There were seventeen (17) team members in the project; project management team (2), consultant team (3), technical team (6), quality assurance team (4), support team (1) and user interface design team (1). These team members have been involved in multiple successful projects within the company. For example, the consultancy team was concurrently engaging in another ISD project, which was categorized as a successful project. Thus, the team members were capable of planning, analysing and delivering successful ISD projects. This project had a client company from Asian region which provided brokerage services for stock exchanges. The client company commenced its operations in 2000s with the intension of becoming a leading financial intermediary for providing capital market access to investors. The company had a wide range of clients including financial institutions, corporates and banks. The brokering products of the company included equity, derivative and on-line trading products. Client company dealt with multiple exchanges, which had multiple asset classes such as equities, securities lending and borrowing[3]. Each asset class consisted of different market types such as normal markets and auction markets.

Project followed agile methods where the information system development tasks were divided into multiple iterations with short time frames, where at the end of each iteration some proportion of the information system was delivered to the client (Boehm 1988). These iterations involved the team working through a full ISD lifecycle including requirements engineering, design, coding and testing. Multiple iterations were required to release a complete software product for the clients. In the requirements elicitation stage of the project, client requirements were gathered and decomposed. Similar requirements were grouped as eleven (11) requirement modules, which were described in 11 Business Requirement Specifications (BRSs)[4]. BRSs consisted of information about the requirement modules including functionalities, dependencies, parameters and concepts. For example, trade processing BRS explained the procedure of entering a trade in to the system, trade processing, trade management, contract and bill generation, trade confirmation and trade rejection procedures. Each BRS of the project contained around 50-100 pages. As specified in the deliverables dashboard (V1.0) document, requirement modules of the system were planned to be delivered in three deliverables: deliverable 1 - client registration, master data, trade processing and user management; deliverable 2 – fund processing, stock processing, brokerage taxes and charges and general accounting and journal

---

[2] To maintain confidentiality, the name of the company, project and personnel were disguised.

[3] For a wider discussion of the above concepts see for example Senarath and Copp (2015), Senarath (2016) and Senarath (2017)

[4] BRSs are also referred to as software requirement specifications, functional specifications, product specifications, system specifications or requirement documents

entries; and deliverable 3 – derivatives, initial public offering and manual funds processing and depository participant module. Since trade processing, fund processing and stock processing were identified as main functionalities of the system, this study mainly focused on these three BRSs. During these deliverables, multiple iterations of requirements engineering process was conducted. Since consultants of the failed project documented requirements while analysing requirements, we integrated the two steps (i.e. requirement analysis and modelling step and requirement communication step) suggested by Nuseibeh and Easterbrook (2000) in to one for analysis purposes (see figure 1). As depicted in figure 1; evolving requirements triggered a new requirement engineering process iteration.
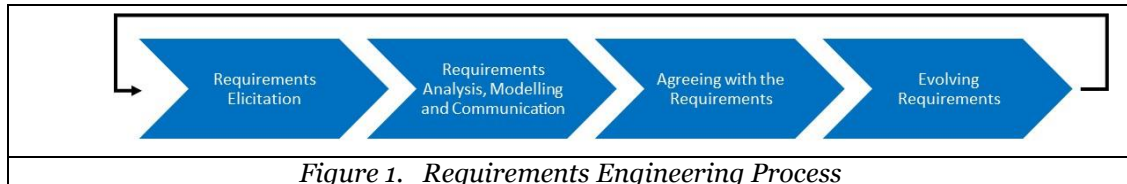

Figure 1.   Requirements Engineering Process

3.2   Data Collection and Analysis

Six (6) semi-structured interviews, lasting between 30-40 minutes each, were conducted with team members of consultant team, technical team, project management team and quality assurance team. The sampling technique is non-probability, purposive and employed the 'snowball technique', where the interview participants were appropriate opinion leaders with well-developed views on the research topic (Minichiello et al. 1995). At the beginning of each interview, the participants were briefed about objectives of the study. All the interviews were recorded, with notes taken when necessary, and then transcribed. In total, our interviews of approximately 3 ½ hours transcribed into 45 pages. The interviews were supplemented with the BRSs and other internal documents such as process flow diagrams and mind maps. Eleven (11) BRSs, five (5) process flow diagrams, two (2) mind maps and one (1) deliverables dashboard document were collected and analysed. The BRS is considered as the contract which specifies the client requirements whereas the process flow diagrams and mind maps indicate the relationships between requirements and the flow of requirement execution. Deliverable dashboard document was used to outline the expected deliverables of the project. Grinnell, Gabor, and Unrau (2015) recommended the use of objective data (e.g. BRSs and process flow diagrams) to reduce the ambiguity of data and generate unbiased results.

Predominantly a deductive approach was adopted in this research. Coding guidelines for control mechanisms were informed by the literature. For example Choudhury and Sabherwal (2003, p. 301) discussed that "mechanisms to explicitly specify desired outcomes that were assessed later" can be identified as outcome controls. Therefore, "expected outcomes are written in detail in the BRSs" was identified as one characteristic of formal-outcome controls. Control mechanisms were established as follows; 1) formal outcome controls - expected outcomes of the project were identified and documented in the BRSs (i.e. client requirements and system functionalities), 2) formal behaviour controls - expected behaviours of the project were identified and documented in the BRSs (i.e. steps that should be followed to achieve the expected outcomes), 3) informal-clan controls - shared values and beliefs, team spirit, interactions and collaborations between team members, and 4) informal self-controls - ISD team members' ability to control their own actions, provide feedback and suggestions and execute the project as per their requirements.

4   Results

Initially, few team members of the project (i.e. a consultant, a project manager and a tech lead) visited the client premises to conduct initial discussions. During initial discussions, the team and the client were able to agree with the high-level project goals, timeline and budget. After returning, the consultant team commenced requirement analysis, modelling and documentation. During this stage, consultants created process flow diagrams to explain the process flows of main functionalities of Information System (i.e. trade processing, fund processing and stock processing) and commenced writing BRSs. Consultant team had several conference calls with the client to get the requirement clarifications. Draft BRSs, which were documented by the consultants were sent for client review. First iteration of requirements engineering process completed in November 2010 with following outputs: Trade processing BRS (V1.00), Fund processing BRS (V1.00) and Stock processing BRS (V1.00). The signed-off BRSs were shared with the technical team for development.

Second round of requirements engineering process was supported by the signed-off BRSs of the first round. After the second round of requirement elicitation, in January 2011, the initial BRSs of trade processing, fund processing and stock processing were updated with client comments. For example, in revision history of trade processing BRS, it was mentioned *"Jan 21, 2011 - Trade Processing (V1.01) - Updated with [client] comments (Entire document updated)"*. Second iteration of requirements engineering process was completed on January 2011 with following outputs: Trade processing BRS (V1.01), Fund processing BRS (V1.01), Stock processing BRS (V1.01), trade processing flow diagram (V1.00), fund processing flow diagram and stock processing flow diagram (V1.01). During the third iteration, a lead consultant of the project visited client site again for one month. During this period, consultant continued requirement elicitation, analysis and BRS documentation. In the revision history of trade processing BRS it was mentioned; *"March 10, 2011 – Trade Processing (V1.02) Updated spec [specification] with feedback received from the visit to [client]"*. When the project goes through multiple iterations, BRSs were updated including more information. For example, trade processing BRS of the project had eight (8) main versions. Although there were only 16 pages in trade processing BRS (V1.00), trade processing BRS (V1.08) consisted of 154 pages. The requirements iterations took place for over three years and eventually the project was abandoned. Respondent 02 mentioned; *"It is almost at the exit procedure level. So, we were [in] discussion how we get out from the project"*. Respondent 01 mentioned; *"We have temporary halted it, but we haven't stopped it. We are looking for another potential client. If we get one, we will open it again"*.

Controls in requirements engineering process iterations:

During the initial stages, draft BRSs consisted of broad information on expected outcomes and behaviours of Information System to be developed. Thus, the level of outcome and behaviour controls were recognized as low. When the project goes through multiple iterations, BRSs were updated including more information. Respondent 01 explained; *"For like two years the BRS have been frequently been updated. So, after like two years we had 95% complete set of BRSs. But, initially that was not completed"*. This indicates that the level of outcome and behaviour controls were enhanced gradually. Since the discussions between client and consult team was friendly during initial agreements, the level of clan control between client and consultant team was identified as high. Since the consultants had the ability to suggest and discuss functionalities of the Information System, the level of self-control of the consultant team was identified as high. However, the level of clan control between client and consultant team declined gradually. Respondent 02 mentioned; *"The client [is] so demanding, whatever requirement they find out today, they put in to the go live scope. They say ok, this should be there, it is obvious thing it should be there"*. One member from the technical staff joined the team for initial client visit. Other than that, there was minimum involvement of technical staff during requirements engineering process. Lack of involvement and collaborations between consultants and technical team indicated low level of clan control between consultant team and technical team. As a result of lack of collaborations, the technical team was unable to suggest and discuss functionalities of the system. Thus, the level of self-control of the technical team was identified as low.

Key issues of the control portfolio as per the four stages of requirements engineering process are described below.

Step 1: Requirements elicitation

Lead consultant went to the client site for one month of requirement elicitation. However, one month of requirement gathering at the client site was insufficient due to the large scope of the project. Respondent 01 explained; *"Problem with the requirement gathering was in the initial part. The time given to her was a month. [..] One month is not enough. [..] Scope is very large. [..] There were business requirement specification after that month, but when the project progress we realized [..] the requirement is not very clear"*. This highlights that expected outcomes and behaviours of Information System were not clearly identified during requirement elicitation stage. During the project execution, client was required to be contacted for requirement clarifications. Since the client was not committed for project, consultant team did not receive answers for their queries. *"Their [clients'] commitment was minimum. So, when we even ask a simple question they didn't reply"* [respondent 01]. As a result, consultant team had to find alternative methods to elaborate client requirements. Respondent 01 mentioned; *"It was very difficult, sometimes we came up with solutions, couldn't wait for the client reply. We have to come up with the solutions and we have to elaborate on the functionalities"*. Due to lack of client commitment, consultant team defined expected outcomes and behaviours of the system as per their knowledge and understanding of client requirements. Some of the software functionalities suggested by the consultant team were never used by the client. *"Since we don't know the exact way that they are doing that functionality we just try to come up with several alternatives which would*

never be used by the client. So, that was a big problem" [respondent 01]. With the intension of reselling the information system to other potential clients, the consultant team included new requirements even without client request. Respondent 01 explained; *"Most of the time what we did was we added some alternatives. For example, if the client wants one and two [features] we added three and four".* This indicated inaccurate estimations of outcome and behaviour controls of the system.

Step 2: Requirements analysis, modelling and communication

Signed BRSs consisted of high-level functional requirements. There was lack of information about the implementation procedures. *"They signed-off the business functionality basically. BRSs don't have this is exactly how we are going to give [this] to you"* [respondent 01]. Since the BRSs did not include detail information about the requirements, quality assurance team encountered difficulties during testing. Respondent 06 explained; *"Sometimes we don't know whether it is defect or not. We don't have proper BRSs. That is the major problem".* This highlights that lack of outcome and behaviour controls originated issues in testing stage. After the BRSs were agreed and signed-off by the clients, consultants updated BRSs including implementation details. However, the client was not informed about these updates. According to respondent 01; *"We did the changes on the top of signed BRSs. [..] We couldn't update the client. So, the signed-off BRSs are like one set, we have a new set of BRSs which are something different from the signed-off BRSs".* This highlighted that the consultant team had high level of self-control, where they updated the signed-off BRSs as per the requirements of the team without client approval. When the consultants suggested a solution for a specific issue in the requirements or else an ISD implementation method, the technical team did not agree with the consultant's suggestions. Respondent 01 discussed: *"Most of the time, when we [consultants] suggest a problem or suggest a solution [technical team mention that] we can't do this".* This highlights that there was lack of clan control between consultant team and technical team.

Step 3: Agreeing with the requirements

Draft BRSs were not reviewed by the technical team before signing off. Respondent 01 explained*; "We should have given the BRSs before signed off for the development to review. It never happened".* As a result, technical team was unable to provide information about the technical limitations of the system. Thus, the level of self-control of the technical team was identified as low. When the project goes through multiple iterations, agreeing with the requirements became much difficult. There were many disagreements between clients and consultant team. This highlights that the level of clan control between client and consultant team declined gradually. For example, on 20th September 2011, around fifty (50) spec points5 of trade processing BRS was updated as per the client requests during BRS finalization calls.

Step 4: Evolving requirements

Since the signed-off BRSs did not include clear information, the client requested several new requirements after the BRSs were signed-off. Respondent 02 discussed; *"The client [is] so demanding, whatever requirement they find out today, they put in to the go live scope. They say ok, this should be there, it is obvious thing it should be there, initially we didn't identify, didn't realize [the difficulty of providing the functionalities]".* Since some of the initially agreed requirements could not be implemented because of the interdependencies, those requirements were removed later in the ISD lifecycle. Respondent 01 from stated: *"This requirement cannot be implemented without that [requirement], because it is clashed with other requirement. So, a big requirement was removed".* Since the BRSs were updated after removing the functionalities, the software engineers had to amend the software code accordingly. Respondent 02 stated: *"They [software engineers] have to change certain things, because the document [BRS] is changing, it is changing continuously. It is frequently changing. So, the developer [software engineers] can't always accommodate the changes [...] It is not easy. It is not the proper practice".* This highlights the volatile nature of expected outcomes and behaviours of the system.

Table 1 summarizes the key issues of the control portfolio as per the four stages of requirements engineering process.

---

5 Spec point- a description written under a specific number in BRS (e.g. section 2.2.1 – update trade postings).

Table 1: Summary of Key Issues of the Control Portfolio

Stage	Description	Key issues of control portfolio
Requirement elicitation	Requirements are not clear	Expected outcomes and behaviours of Information System were not clearly identified
	Some requirements suggested by consultants were never used by clients	Inaccurate estimations of expected outcomes and behaviours of the system Domination of self-control of consultants
	Consultant team included new requirements even without client request	Inaccurate estimations of expected outcomes and behaviours of the system Domination of self-control of consultants
Requirements analysis modelling and communication	Lack of information in BRSs	Expected outcomes and behaviours of Information System were not properly documented
	Documented BRSs were updated many times including clarification information	Expected outcomes and behaviours of Information System were not properly documented
	Consultants updated signed-off BRSs	Domination of self-control of consultants
	Technical team disagree with the consultants' suggestions	lack of clan control between consultant team and technical team
Agreeing with the requirements	BRSs were not reviewed by the technical staff before sign-off	Low level of technical team self-controls
	Many disagreements during BRS finalization calls	Level of clan control between client and consultant team declined gradually
Evolving requirements	Client requested several new requirements after the BRSs were signed-off	Volatile outcome and behaviour controls
	Some of the requirements were removed from the BRSs	Volatile outcome and behaviour controls

5   Discussion

This study was motivated by the need to understand the how the weaknesses in the control portfolio contributed to an ISD-outsourcing project failure. Analysis of interview data through the control theory perspective provided the ability to understand this phenomenon. Control portfolios were visible in all stages of requirements engineering process. Therefore, we attribute the failure not to the absence of control portfolios, rather (i) employing particular control mechanisms **incorrectly** depending on the phase of requirements engineering process, (ii) **dominance** of a particular control mechanism, and (iii) **insufficiency** of particular control mechanisms. For example, some of the software functionalities suggested by the consultant team were never used by the client. This indicates incorrect estimations of expected outcomes and behaviours of the system during requirement elicitation stage. As per Guinan et al. (1998), some consultants believe that they are aware of what is best for the clients and are not willing to work with requirements provided by the clients. With the intension of reselling the information system to other potential clients, the consultant team included new requirements even without client request. Some software development projects over-heat to failures as a result of ambitious consultants overpromising the software deliverables (Zhu 2012). This indicates dominance of consultants' self-controls during requirement elicitation stage. BRSs were documented during requirement analysis, modelling and communication stage. These BRSs did not include sufficient information about the expected outcomes and behaviours of information system to be developed. This indicates insufficiency of outcomes and behaviour controls. Previous research (Karlsson et al. 2007; Ricca et al. 2009) highlighted similar concerns of unclear and incomplete information about expected outcomes and behaviours of information systems.

Findings of the study highlighted the importance of further exploring the control portfolios focusing on the stages of requirements engineering process. Findings of this study have the potential to influence practice. First, the findings help ISD-outsourcing companies to understand how the incorrect execution of control mechanisms, dominance of particular control mechanisms or insufficiency of control mechanisms could lead to project failures. Second, it highlights the importance of focusing on control mechanisms as per the stages of requirements engineering process. Third, the application of control theory perspective provides useful guidelines for ISD-outsourcing, which could ultimately minimize ISD-outsourcing project failures. Several studies in the literature discussed stages of requirements engineering (Nuseibeh and Easterbrook 2000; Pohl 1996) and application of control mechanisms in ISD-outsourcing projects (Choudhury and Sabherwal 2003; Nuwangi et al. 2018); however, this study provides a novel perspective of identifying appropriate control mechanisms for ISD-outsourcing projects as per the stages of requirements engineering process. Through the control theory perspective, this research highlighted that incorrect execution of control mechanisms, dominance of particular control mechanisms or insufficiency of control mechanisms could lead to ISD-outsourcing project failures.

6 References

Abrahamsson, P., Salo, O., Ronkainen, J., and Warsta, J. 2017. "Agile Software Development Methods: Review and Analysis," *VTT Technical Research Center*), pp. 1-107.

Boehm, B. W. 1988. "A Spiral Model of Software Development and Enhancement," *Computer Modelling and Simulation of Smart and Green Computing Systems* (21:5), pp. 61-72.

Choudhury, V., and Sabherwal, R. 2003. "Portfolios of Control in Outsourced Software Development Projects," *Information Systems Research* (14:3), pp. 291-314.

Chua, A., and Lam, W. 2005. "Why KM Projects Fail: A Multi-Case Analysis," *Journal of knowledge management* (9:3), pp. 6-17.

Deloitte. 2016. "Deloitte's 2016 Global Outsourcing Survey."

Ehrhart, T. 2002. "All Wound Up: Avoiding Broken Promises in Technology Projects," *Risk Management* (49:7), pp. 12-17.

Gartner. 2017. "Market Share Analysis: IT Outsourcing Services, Worldwide, 2016." from https://www.gartner.com/doc/3744417/market-share-analysis-it-outsourcing

Gregory, R. W., Beck, R., and Keil, M. 2013. "Control Balancing in Information Systems Development Offshoring Projects," *MIS Quarterly* (37:4), pp. 1211-1232.

Guinan, P. J., Cooprider, J. G., and Faraj, S. 1998. "Enabling Software Development Team Performance During Requirements Definition: A Behavioral Versus Technical Approach," *Information systems research* (9:2), pp. 101-125.

Hastie, S., and Wojewod, S. 2015. "Standish Group 2015 Chaos Report." from http://www.infoq.com/articles/standish-chaos-2015

Karlsson, L., Dahlstedt, Å. G., Regnell, B., och Dag, J. N., and Persson, A. 2007. "Requirements Engineering Challenges in Market-Driven Software Development–an Interview Study with Practitioners," *Information and Software technology* (49:6), pp. 588-604.

Khan, S. U., Niazi, M., and Ahmad, R. 2011. "Barriers in the Selection of Offshore Software Development Outsourcing Vendors: An Exploratory Study Using a Systematic Literature Review," *Information and Software Technology* (53:7), pp. 693-706.

Kirsch, L. J. 1996. "The Management of Complex Tasks in Organizations: Controlling the Systems Development Process," *Organization Science* (7:1), pp. 1-21.

Lyytinen, K., and Robey, D. 1999. "Learning Failure in Information Systems Development," *Information Systems Journal* (9:2), pp. 85-101.

Minichiello, V., Aroni, R., Timewell, E., and Alexander, L. 1995. *In-Depth Interviewing: Principles, Techniques, Analysis*, (2nd ed.). Melbourne, Australia: Pearson Education Australia.

Mishra, D., Mishra, A., and Yazici, A. 2008. "Successful Requirement Elicitation by Combining Requirement Engineering Techniques," *First International Conference on the Applications of Digital Information and Web Technologies*: IEEE Xplore Digital Library, pp. 258-263.

Nakatsu, R. T., and Iacovou, C. L. 2009. "A Comparative Study of Important Risk Factors Involved in Offshore and Domestic Outsourcing of Software Development Projects: A Two-Panel Delphi Study," *Information & Management* (46:1), pp. 57-68.

Nuseibeh, B., and Easterbrook, S. 2000. "Requirements Engineering: A Roadmap," *Conference on the Future of Software Engineering*: ACM Digital Library, pp. 35-46.

Nuwangi, S. M. 2016. "The Impact of Modularisation on Information System Development Outsourcing Project Control," in: *Information Systems*. Queensland University of Technology, Australia p. 259.

Nuwangi, S. M., and Sedera, D. 2017. "The Impact of over Ambitious Consultants on Project Failures: An Ambition Theory View," in: *ANZAM Conference*. Melbourne, Australia: ANZAM.

Nuwangi, S. M., Sedera, D., and Srivastava, S. C. 2014. "Introducing System Controls for Control Theory," *Australasian Conference on Information Systems*, Auckland, New Zealand.

Nuwangi, S. M., Sedera, D., and Srivastava, S. C. 2018. "Multi-Layered Control Mechanisms in Software Development Outsourcing," *Pacific Asia Conference on Information Systems*, Yokohama, Japan: Association for Information Systems AIS Electronic Library (AISeL), pp. 1-9.

Ouchi, W. G. 1978. "The Transmission of Control through Organizational Hierarchy," *Academy of Management Journal* (21:2), pp. 173-192.

Paetsch, F., Eberlein, A., and Maurer, F. 2003. "Requirements Engineering and Agile Software Development," *Enabling Technologies: Infrastructure for Collaborative Enterprises*: IEEE Xplore Digital Library, pp. 308-313.

Pandey, D., Suman, U., and Ramani, A. 2010. "An Effective Requirement Engineering Process Model for Software Development and Requirements Management," *International conference on Advances in recent technologies in communication and computing*: IEEE, pp. 287-291.

Pee, L. G., Kankanhalli, A., and Kim, H.-W. 2010. "Knowledge Sharing in Information Systems Development: A Social Interdependence Perspective," *Journal of the Association for Information Systems* (11:10), pp. 550 - 575.

Pohl, K. 1996. *Process-Centered Requirements Engineering*. New York, USA: John Wiley & Sons, Inc.

Ricca, F., Torchiano, M., Di Penta, M., Ceccato, M., and Tonella, P. 2009. "Using Acceptance Tests as a Support for Clarifying Requirements: A Series of Experiments," *Information and Software Technology* (51:2), pp. 270-283.

Saiedian, H., and Dale, R. 2000. "Requirements Engineering: Making the Connection between the Software Developer and Customer," *Information and Software Technology* (42:6), pp. 419-428.

Salado, A., and Nilchiani, R. 2016. "The Concept of Order of Conflict in Requirements Engineering," *IEEE Systems Journal* (10:1), pp. 25-35.

Savolainen, P., Ahonen, J. J., and Richardson, I. 2012. "Software Development Project Success and Failure from the Supplier's Perspective: A Systematic Literature Review," *International Journal of Project Management* (30:4), pp. 458-469.

Sedera, D., Lokuge, S., Krcmar, H., Srivastava, S. C., and Ravishankar, M. N. 2014. "The Future of Outsourcing in the Asia-Pacific Region: Implications for Research and Practice—Panel Report from PACIS 2014," *Communications of the Association for Information Systems* (35:1), pp. 317-331.

Senarath, S. 2016. "Not So 'Bankruptcy-Remote': An Insight into Sri Lankan Securitization Practices in a Post_Gfc Context," in: *Multidisciplinary Academic Conference on Management, Marketing and Economics*. pp. 53-60.

Senarath, S. 2017. "The Dodd-Frank Act Doesn't Solve the Principal-Agent Problem in Asset Securitisation." LSE Research Online: LSE Business Review.

Senarath, S., and Copp, R. 2015. "Credit Default Swaps and the Global Financial Crisis: Reframing Credit Default Swaps as Quasi-Insurance," *Global Economy and Finance Journal* (8:1), pp. 135-149.

Sim, W. W., and Brouse, P. S. 2014. "Empowering Requirements Engineering Activities with Personas," *Procedia Computer Science* (28), pp. 237-246.

Srivastava, S. C., and Teo, T. S. H. 2012. "Contract Performance in Offshore Systems Development: Role of Control Mechanisms," *Journal of Management Information Systems* (29:1), pp. 115-158.

Standish Group. 2014. "The Standish Group Report: Chaos."

Tiwana, A. 2004. "An Empirical Study of the Effect of Knowledge Integration on Software Development Performance," *Information and Software Technology* (46:13), pp. 899-906.

Tiwari, S., Rathore, S. S., and Gupta, A. 2012. "Selecting Requirement Elicitation Techniques for Software Projects," *International Conference on Software Engineering (CONSEG)*: IEEE Xplore Digital Library, pp. 1-10.

Wiener, M., Mähring, M., Remus, U., and Saunders, C. 2016. "Control Configuration and Control Enactment in Information Systems Projects: Review and Expanded Theoretical Framework," *MIS Quarterly* (40:3), pp. 741-774.

Yin, R. K. 2009. *Case Study Research: Design and Methods* (4th ed.). Thousand Oaks, CA: Sage Publications.

Zhu, P. 2012. "Five Ponderings on Why IT Projects Fail." from http://futureofcio.blogspot.com.au/2012/01/five-pondering-why-it-projects-fail.html