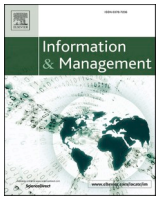
Information & Management 61 (2024) 103962

Contents lists available at ScienceDirect

Information & Management

journal homepage: www.elsevier.com/locate/im



Understanding the nature of information interdependencies and developing control portfolios for modularized information systems development projects

Maduka Subasinghage [a], Darshana Sedera [b,*], Shirish C. Srivastava [c]

[a] Business Information Systems Department, Auckland University of Technology, Auckland, New Zealand
[b] Southern Cross University, Gold Coast, Australia
[c] HEC Paris, 1 Rue de la Libération, 78351 Jouy en Josas Cedex, France

ARTICLE INFO

Keywords:
Information interdependencies
Modularization
Control theory
Information systems development projects
Requirements
Governance

ABSTRACT

For better governance, information systems development (ISD) projects are often decomposed ex ante, using client requirements, to create self-contained modules. Following such a modular approach allows independent and efficient structuring of ISD project work. However, given the integrated nature of ISD work, maintaining information flows across the different modules within ISD projects is essential. Through an in-depth examination of six ISD projects, we discover four types of requirements information interdependencies across ISD project modules. Next, we identify appropriate control portfolios for governing the different types of inter-modular requirements information interdependencies. Results indicate that different types of requirements information interdependencies necessitate governance through varying control portfolios. Our study contributes to the ISD literature by identifying the nuanced mechanisms through which requirements information interdependencies could be better governed by applying appropriate control portfolios. Our study also offers significant practical implications for ISD project governance.

1. Introduction

Modularization is a technique commonly employed in information systems development (ISD) projects to make project governance more efficient by decomposing the project into self-contained modules, which are relatively less complex and purportedly easier to manage [1]. Though prior ISD research has studied modularization of ISD projects, the focus has largely been on the modalities for splitting the project into self-contained modules and governing them independently [2]. Though modularization is expected to segregate the ISD project into independent work units, given the integrated nature of ISD work, there may still be multifarious interdependencies across the seemingly independent modules [3,4]. Thus, the ability of ISD projects to achieve the expected outcomes is dependent on how efficiently the various types of inter-modular interdependencies are governed. Researchers recognize various types of interdependencies, ranging from organizational [5,6] and project [7,8] to teams [9,10]. This research focuses solely on the *specified* information interdependencies between modules in an ISD project. To govern these information interdependencies efficiently, it is important to understand not only the nature of the various information interdependencies, but also appropriate governance mechanisms for managing them. Hence, research that examines the mechanisms through which the different types of information interdependencies can be better governed will be of value for both theory and practice.

ISD projects generally unfold in three phases: (1) requirements elicitation and design phase, (2) development phase, and (3) testing and implementation phase [11]. During phase 1, the business analysts identify and classify client requirements with functional similarities to create requirements modules. The client requirements belonging to each module are described in a business requirements specification (BRS) document. Once the business requirements are established, the software engineers develop modules in phase 2 per the description in the BRSs. During phase 3, software and quality assurance engineers test the developed modules. Accordingly, BRSs are a prime source of documentation for identifying the nature of the requirements information interdependencies (henceforth referred to as *information interdependencies*) across different modules in an ISD project.

Managing ISD projects with information interdependencies across

* Corresponding author.
E-mail address: darshana.sedera@gmail.com (D. Sedera).

https://doi.org/10.1016/j.im.2024.103962
Received 9 June 2021; Received in revised form 3 April 2024; Accepted 7 April 2024
Available online 9 April 2024

modules is innately complex [12]. Considering that a team assigned to one module focuses predominantly on developing the assigned module, rather than on the integration between related modules, information interdependencies between modules cause iterative and interdependent complexities. To better govern such information interdependencies, project managers need to be cognizant of the different types of information interdependencies across modules. This will help in formulating appropriate control mechanisms for governing ISD projects to achieve the expected goals [13].

In this research, we aim to examine how ISD projects with different types of information interdependencies can be governed efficiently. Specifically, to identify the appropriate governance mechanisms, we ground our research in control theory [14,15], which has been found to be a useful theoretical lens for understanding ISD project governance in different contexts [16–18]. Accordingly, the two questions that we address in this research are as follows:

RQ1 – What are the different types of information interdependencies in modularized ISD projects?

RQ2 – How can a control portfolio be developed based on the types of information interdependencies identified in RQ1, so that the ISD projects can be governed effectively? Herein, we use the term "Control portfolio" to denote various configurations of controls, simultaneously configured and re-configured using the outcome, behavior, clan, and self-control modes [19].

To answer these questions, we leverage data from six ISD projects implemented by a large IT firm that specializes in developing capital market solutions. The data for our study comprise detailed project governance documentation (e.g., BRS documents, project description documents, change request documents, and software design documents), onsite observations, and interviews with project members from the sampled ISD projects. First, we *inductively* determine the different types of information interdependencies, considering the information flows between modules within each ISD project. Thereafter, we *deductively* delineate the corresponding control portfolios employed and the performance of each of the examined ISD projects.It is important to identify the project performance, so that we can check which control portfolio aligns best with a particular type of information interdependency.

Our research makes three primary contributions: (1) We extend prior research on modularization [3,20] by better explaining the modularization conducted during "requirements elicitation phase" in the ISD project life cycle, (2) we examine the types of information interdependencies per the information flow between modules, which may be the key to better project governance and output quality, and (3) we extend prior research on control theory [21,22] and contribute to a better understanding of how control portfolios can be derived and used for governing modularized ISD projects with information interdependencies. Though there is a substantial body of work on ISD project governance in general (see, for example, Wiener, Mähring, Remus, and Saunders [23], Cecil and Myers [24], and Maruping, Venkatesh, and Agarwal [25]), this research focuses on the nuanced view of how ISD projects are governed *based on the types of information interdependencies* that are observed through the study. Our research builds on and contributes to the prior research on ISD project governance, such as Tiwana [2], which postulates the importance of examining modular interdependencies.

2. Theoretical background

This section explains the interdependent modules and the role of control theory in governing information interdependencies between modules. The first subsection explains the modularization and information interdependencies, followed by the second, which explains the types of information interdependencies. The third explains the use of control theory in managing the information interdependencies. Finally, the fourth establishes the research gap and explains the focus of our study.

2.1. Modularization and information interdependencies

Hölttä-Otto and de Weck [[26], p. 113] define a module as "an independent chunk that is highly coupled within, but only loosely coupled to the rest of the system." Mikkola and Skjøtt-Larsen [[27], p. 354] define modularization as the process of "decomposing complex tasks" into "simpler portions" so that they can be "managed independently" and "yet operate together as a whole," highlighting the essential attributes of modularization. Therefore, modularization puts a high level of emphasis on creating modules that are mutually exclusive and independent, minimizing information interdependencies. Minimizing the interdependencies between modules reduces the coordination costs [28,29]. Thus, many researchers [30,31] highlight the importance of reducing interdependencies between modules. However, modules in an ISD context are rarely fully independent. Driven by the necessity of creating an integrated information system, even if a module (i.e., a software function) can be developed in isolation, such functions must be integrated with other parts of the broad system. Therefore, information interdependencies between modules in an ISD project are inevitable [4]. Such information interdependencies, in the development phase, occur as "information flows" between modules where, for example, module A passes information to module B and vice versa (see Fig. 1). As a result of these information interdependencies, an update in the requirements of module A at the time of developing the software will result in changes to module B as well (and vice versa). These recursive information flows between modules occur continuously in the development phase of ISD projects. As a result, ISD projects might not be able to meet the expected project goals within the agreed time frames. Thus, it is important that project managers consider the nature of information flows between requirement modules when deciding on control mechanisms for ISD projects.

With a broader connotation, each module in Fig. 1 can be treated as an IT artefact. Characteristics of each module in an ISD project are consistent with the Riemer & Johnston [32] definition, where an IT artefact is defined as "a bundle of features or properties" ([32], p. 273). Thus, each module presents similar features, and properties of each artefact (i.e., module) are described in a detailed document called a business requirements specification (BRS) document.

2.2. Types and entities of information interdependencies

In considering information interdependencies, one could pair types with entities of information interdependencies. Previous research [4,5] discusses various *types of interdependencies* (e.g., task information interdependencies, goal information interdependencies, development information interdependencies) and entities of interdependencies (e.g., inter-/intra-organizational). The task interdependencies and goal interdependencies originate interactions between ISD team members assigned to different modules [5]. There are development interdependencies (unplanned interactions occur between software development tasks belong to different modules) and developer interdependencies (interactions between software developers assigned to different modules) in ISD projects [4]. In relation to the *entities of interdependencies*, there is substantial research on inter-/intra-organizational information interdependencies [5,6]. The conflicts

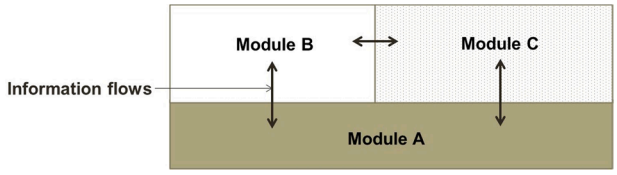


Fig. 1. Information interdependencies between modules.

that might occur due to organizational interdependencies have to be managed through various governance mechanisms [33,34]. Using ISD projects as examples, Ganesh and Gupta [35] explained that when there are task interdependencies between team members, they cooperate with each other to complete the tasks on time. This research extends the existing literature [12,36,37] by identifying the *information interdependency types* based on the *nature of information flows between modules* (i.e., entities) *in ISD projects*. We acknowledge that there may be some inter-relationships between one entity of interdependency (e.g., organizational with module) and another. Yet, for the focus of this research, such inter-relationships are ignored and only the inter-modular information interdependencies are explored. Further details about the types of interdependencies are provided in Appendix A.

2.3. Managing information interdependencies using control theory

While modularization reduces complexity, researchers argue that interdependencies could negatively impact project performance [38]. For example, in ISD projects, the client requirements are modularized, and ISD team members are assigned to different modules. When there are many information interdependencies between modules, the team members in one module must frequently communicate with the team members of other modules to maintain the integrated nature of information systems, which ultimately minimizes the project performance. On the other hand, when there is a lack of information interdependencies between modules, the team members tend to work independently, with minimum communication with the team members allocated to other modules. However, the team members from different modules must ensure that the developed information systems are well integrated to execute the functionalities. Thus, it is important to understand the nature of information interdependencies and develop control portfolios to govern team members assigned to different modules to minimize such issues in ISD projects.

Carlshamre, Sandahl, Lindvall, Regnell, and Dag [39] argue that the focus should be on identifying a few interdependencies that account for most of the actual intra-project interdependencies and place maximum effort on managing team members assigned such significant interdependencies. Overall, there is a broader consensus that interdependencies within ISD projects are complex due to the tensions between autonomy within a module and the integration sought between the modules, making project management a challenging task. Appendix B provides a sample of research that consolidates perspectives of interdependencies, modularity, and project governance in the context of ISD projects.

We employ the foundations of control theory [21,22] to understand how information interdependencies are governed using a portfolio of controls so that a project can *achieve its stipulated goals or targets* [14]. More specifically, our research explores how to develop control portfolios to *govern* [40] the team members assigned to different modules, so that *ISD projects can achieve the goal of executing the ISD projects under stable conditions.* Herein, we acknowledge that there are two alternative theories in relation to governance: coordination theory and control theory. According to Sabherwal ([41], p. 155), "coordination focuses on managing interdependencies among multiple individuals or activities involved in the overall task." Kirsch ([14], p. 215) defined control as "encompassing all attempts to ensure individuals in organizations act in a manner that is consistent with meeting organizational goals and objectives." Rather than focusing only on coordination, this paper aims to explore how to develop control portfolios to govern the information interdependencies, so that ISD projects can achieve the goal of executing projects under stable conditions; thus control theory is considered most appropriate for our study. Given the breadth of research on control theory applications in deriving governance mechanisms in a wide range of complex contexts such as e-commerce [42], organizational politics [43], and enterprise resource planning projects [44], its selection as the theoretical lens for our work is further justified. Moreover, control theory is one of the most prominent theoretical foundations that have been employed to study ISD projects; see, for example, Cecil and Myers [24], Keil, Rai, and Liu [45], Maruping et al. [25], Cram and Brohman [46], and Wiener et al. [23]. Scholars have extended control theory by introducing concepts such as control styles (e.g., enabling/-authoritative), control degree (e.g., frequency/intensity), control types (e.g., social type/procedural), and technology-mediated control (see examples in [16,24,42,47]). Rather than focusing on extending control theory, we apply the basic tenets of the theory to explain how to develop control portfolios to govern ISD projects with inter-modular requirements interdependencies.

Control theory literature outlines four control modes, namely *formal–outcome control, formal–behavior control, informal–clan control,* and *informal–self-control* [23,48,49]. Formal control involves governing employees through mechanisms such as performance evaluations, where either the outcomes or behaviors of employees are measured, evaluated, and rewarded [48]. Formal control mechanisms are further subdivided into outcome-based and behavior-based control modes, where outcome-based modes specify the expected outcomes of projects and behavior-based modes specify expected behavior [50,51]. Common project governance techniques that firms employ in ISD projects, such as BRSs, project plans, and regular meetings, fall under the outcome and behavior control modes [52]. Informal controls such as social and people-based strategies are also commonly employed in ISD projects through clan control or self-control. Ouchi [53] describes clan control as a control mode that promotes common values and beliefs within a clan, that is, 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 [54] in disparate projects. On the other hand, self-control occurs when the employees of a firm self-manage their actions [55]. Researchers such as Cram, Brohman, and Gallupe [22] and Cram, Brohman, and Gallupe [21] argue that a control portfolio must be developed with an understanding of the specific context in which the control mechanisms need to be applied. We argue that a better understanding of information interdependencies between modules is essential to develop appropriate control portfolios for governing modularized ISD projects.

As a result of creating projects with minimal information interdependencies between modules, processes can be standardized and outcomes can be predicted well [56]. Thus, team members assigned to projects with minimal information interdependencies across constituting modules can be governed better by outcome and behavior control modes [57]. However, information interdependencies between modules are needed to maintain the integrative nature warranted in contemporary software [58]. Information interdependencies across modules lead to sharing of resources and work and management practices, which often results in conflicts and incongruous goals [59]. These conflicts make it difficult to implement outcome and behavior control modes in such ISD projects.

When an ISD project has modules with information interdependencies, an update in one module can create updates in the others [60], increasing the uncertainty in projects [61]. In such projects, having a clan control mode can perhaps ensure success because team members can work collaboratively and help each other in unexpected situations [62]. Teams in projects that have minimal information interdependencies between modules tend to work independently [63] and have little communication with other teams [64]. Thus, project managers should enforce appropriate control modes to ensure that the tasks completed by a team assigned to a particular module are aligned with other teams' tasks.

Several studies have discussed the choice of control modes as per the task characteristics [19,65]. For example, Kirsch [[14], p. 216] argues that "controllers in lSD settings structure a portfolio of control modes in order to manage the complexities and subtleties of a task that involves people with various knowledge and skills across time." Modularized projects with minimal information interdependencies could employ the

"authoritative control style" [23,47,66], where project managers control team members. Similarly, projects with high level of information interdependencies can be governed better in the "enabling control style" [23]. However, increased homogeneity of a module does not guarantee better outcomes in ISD projects. For example, Sosa, Eppinger, and Rowles [67] argue that minimum interdependencies across modules lead to boundaries between teams, increasing communication barriers between them. In these situations, the project managers should select appropriate control mechanisms to ensure that the developed information systems are well integrated to execute the required functionalities.

2.4. Focus of our study

Previous research [2,15] highlights the importance of considering interdependencies between modules when governing projects. For example, Tiwana [2] explains how minimizing interdependencies between outsourced systems and other software applications replaces control mechanisms. Ouchi and Maguire [15] explain that ISD projects use outcome controls when there are many task dependencies within projects. Tiwana [2] and Tiwana [68] explain how to develop control portfolios considering the level of interdependencies (e.g., high or low). Extending previous research on control theory and interdependencies [2,15,68], this research attempts to explain better how to develop control portfolios based on the *types of information interdependencies* discovered through this research. In situations where project managers have a deeper understanding of the types of information interdependencies, they can establish appropriate control mechanisms to ensure that a project meets the expected goals.

3. Research methodology, data collection, and data sources

We employed a qualitative research design [69], and the case organization was selected using the criterion sampling strategy [70]. Per our sampling strategy, for selecting the research site, we ensured that the selected case ISD organization must (1) modularize its client requirements, (2) be involved in multiple ISD projects to provide adequate access to evidence, and (3) be sufficiently large, with clearly visible roles and responsibilities that enable data collection from multiple employees at various levels of the hierarchy. Following the application of these criteria, StockEX was selected as the case organization. StockEX was established in the year 1996 and currently has over 400 staff members. StockEX specializes in the development of capital market solutions that include the functions of multiple trading methods (e.g., auctions, continuous matching) and multiple asset classes (e.g., equity and fixed income). The reasonably narrow focus of the company is beneficial for our research, as it allows extraneous factors to be controlled for. See Appendix C for a description of the company.

3.1. Data sources

Six (6) ISD projects within StockEX were selected for observation. The selected projects were homogeneous in terms of the industry sector (ISD projects), type of software developed (capital market solutions), and project stage (completed,) but varied in terms of clients (from different countries), team members (different personnel), and outcomes (stable/unstable). At the time of data collection, the selected projects were already completed, which enabled the researchers to determine the information interdependencies, control portfolios, and project durations and witness their degree of stability. See Table 1 for a summary of project descriptions. Note that most of the team members were working in more than one project. Therefore, some projects involved substantial numbers of members. During the data collection, team members were asked to discuss only the selected project.

To assess the information interdependencies of the modules, BRS was employed as the main data source. For example, all six (6) sampled projects consisted of many modules. A BRS was written for each and

Table 1
Summary of project descriptions.

Project pseudonym	Description of the project
A	This project develops a post-stock-trade application, which provides clearing and settlement for trades after execution. Functions of the software application include trade processing, user management, general accounting, and journal entries. Moreover, the software application consists of complex trade processing methods that are highly integrated with clearing and settlement procedures. There are 20 requirements modules in the project. The client is situated in the Asian region. Team members of the project include business analysts (3), software engineers (5), user interface designers (1), system support engineers (1), quality assurance managers (1), project managers (2), director business operations—post-trade (1), assistant vice president software development (1), and line of business manager—post-trade (1).
B	This is a real-time clearing system that manages post-trade activities. This project, which is enriched with the latest technology, provides successful solutions for the limitations of current traditional clearing systems. This project consists of three major functions: (1) clearing (the process of matching, recording, and transaction processing); (2) settlement (trade settlement); and (3) risk management (managing the risks of market participants). There are approximately 21 requirements modules in the project. The client company is situated in Europe. Team members of the project include business analysts (13), software engineers (18), senior user interface designers (1), systems support staff (5), quality assurance engineers (4), project managers (1), director business operations—post-trade (1), line of business manager (1) and product manager—post-trade systems (1).
C	This project addresses a wide range of business needs including connectivity and order management. It can handle a variety of firms' trading business, covering front office, middle office, and back office functions. Key characteristics of the software solution include connectivity hub and post-trade risk management. The project has around 40 clients all over the world, including clients from North America and Asia. The project management team includes business analysts (17), software engineers (29), software support staff (6), quality assurance engineers (5), application operator team (3), project management team (6), director business operations (1), line of business manager (1), product manager (2), and vice president software development (1).
D	This project involves the development of tools to detect irregular trading behavior in a stock exchange. Key functions of the software include analysis of real-time/offline transaction data and providing alerts and case management. This project consists of several clients including clients from Asian and African regions. Team members include business analysts (17), software engineers (24), software support staff (7), quality assurance engineers (5), project managers (6), director business operations (1), project director (1), and vice president software development (2).
E	This project includes the development of a post-trade application to clear and settle executed trades in a stock exchange. There are several users with different authorization levels. The registry owner is the main user. The system has different users (e.g., brokers and custodian banks). There are approximately 10 requirements modules in the project. The client company is situated in the Asian region. Team members of the project include business analysts (5), software engineers (9), systems support staff (1), quality assurance engineers (3), project managers (2), director business operations—post-trade (1), line of business manager—post-trade (1), and assistant vice president software development (1).
F	This project developed a critical application due to client-focused solutions and integrity of modules. The application can be adjusted to trade any product in any type of market. It is a reliable and flexible solution, which meets the client's specific needs. This project has multiple clients all over the world, including clients from Europe. Team members of the project include business analysts (10), software engineers (22), system support engineers (4), quality assurance engineers (4), project managers (3), director business operations (1), and vice president (1).

every module. BRSs consisted of information about the specific modules, including software functions, information dependencies, data parameters, and an explanation of key concepts. For example, the trade-processing BRS of project A explained the procedure of entering trade information into the system and its relationship to the trade processing, trade management, contract and bill generation, trade confirmation, and trade rejection procedures. Each BRS in these projects contained 50–100 pages. The requirements were presented mainly in text format, including diagrams, account postings, and tables, where necessary. BRSs were updated multiple times due to reasons such as project scope refinements. For example, the trade-processing BRS of project A underwent eight version changes. Because each BRS is written to explain a required module, an analysis of BRSs is considered insightful when information interdependencies across modules are explored. Through the analysis of BRSs, we were able to gain a good understanding of the information interdependencies. For example, we were able to understand how information interdependencies occurred, the nature of information interdependencies, the severity of information interdependencies, the sequence of information interdependencies, and so on. We analyzed 43 BRSs, which were linked to project A (14 BRSs), project B (21 BRSs), project E (8 BRSs), projects C (1), project D (1), and project F (1). A sample BRS is illustrated in Appendix D. In addition, nine change request documents were gathered and analyzed from project F, and seven software design documents were analyzed from project C. Through the analysis of change request documents, we were able to understand how the client requirements were updated during the project. Change request documents indicated how the information interdependencies between requirements modules were managed when the client requirements were updated, as well as the challenges of updating client requirements when there are many information interdependencies between requirement modules. Software design documents were helpful for understanding how the software designs were conducted as per the information interdependencies between modules. Project description documents also included information about the requirements modules. Our approach is consistent with the suggestion of Grinnell, Gabor, and Unrau [71], who recommend the use of objective data (e.g., BRSs and project description documents) to reduce data ambiguity and generate unbiased results.

The type of control portfolio was established predominantly using a series of semi-structured interviews. Thus, 17 semi-structured interviews, each lasting approximately 30 min, were conducted with employees from the six ISD projects. See Table 2 for details on participant information.

To avoid key informant bias [72], interviews were conducted with multiple informants from each project. The informant mix consisted of one participant from the consultant team, one participant from the software engineering team, and one participant from the project management team. The sampling of the respondents was non-probabilistic and purposive and employed the snowballing technique [73]. During the semi-structured interviews, we provided closer attention to the types of controls employed and project performance. See Appendix E for the list of guiding interview questions. During the interview, follow-up questions were used to gain in-depth understanding of information interdependencies and control portfolios. All the interviews were audio-recorded and transcribed for subsequent data analysis, yielding approximately 145 pages of single-spaced transcribed documents. While the interviews were conducted, additional notes were taken to supplement the data obtained through the interviews. The interview data were supplemented with project governance documents. These documents were used for data triangulation when information interdependencies and control portfolios were established [74].

Moreover, approximately 40 h of observation of day-to-day operations of the ISD teams (for example, how the software engineers interact with each other, in which situations the software engineers interact with business analysts, and how the business analysts document BRSs) were conducted by the first author. The first author observed team members having casual conversations, projecting related discussions, having lunch together, and so on. Observations were conducted in each project, and during the observations, notes were taken on how the team members assigned to different modules worked together to complete their day-to-day tasks. The observed team members included business analysts, project managers, software engineers, and software quality assurance engineers. Those observations were helpful in understanding the context of ISD projects, the information interdependencies, and control portfolios.

Table 2
Participant information.

Project	Participant ID	Designation	Years of experience in ISD industry	Years of experience in the company
A	01	Senior Business Analyst	4	4
	02	Project Manager	15	2
	03	Specialist Software Engineer	8	8
B	04	Director Business Operations	11	10
	05	Senior Tech Lead	7	7
	06	Senior Business Analyst	4	4
C	07	Business Analyst	4	3
	08	Principal Software Engineer	5	4
	09	Junior Project Manager	4	3
D	10	Senior Software Engineer	8	8
	11	Junior Project Manager	3	3
	12	Senior Business Analyst	3	3
E	13	Specialist Software Engineer	8	8
	14	Consultant	12	10
	02	Project Manager	15	2
F	15	Project Manager	6	5
	16	Senior Software Engineer	3	3
	17	Senior Business Analyst	3	3

4. Data analysis

Consistent with Yin [75], the ISD project was selected as the unit of analysis. The dependability, credibility, transferability, and confirmability of the findings were established during the research (see Appendix F). Data analysis included (1) within-case analysis and (2) cross-case analysis. The overall aim of within-case analysis was to become familiar with each project as a stand-alone entity. Within-case analysis also allows identification of the unique characteristics of six identified ISD projects. This improved the understanding of each project, which ultimately accelerated the subsequent cross-case analysis. Cross-case analysis was conducted to investigate the similarities and differences across the six ISD projects.

Data analysis was conducted following two approaches: while the types of information interdependencies were primarily identified through the BRSs following an inductive approach, the control portfolios and the performance of ISD projects were assessed primarily based on

the employee interviews through a deductive approach. However, both induction and deduction were supported by other sources of data–observation notes, project description documents, change request documents, and software design documents. The knowledge gained through observations was also helpful during the analysis. Please see Fig. 2 for the data analysis process and the key objectives of each phase of analyses.

First, considering the nascent knowledge on the nature of information interdependencies across requirements modules within ISD projects, an inductive approach was adopted for exploring the inherent characteristics of information interdependencies. Open coding, axial coding, and selective coding were performed preliminarily on the BRSs, and were supported by the interview data, observation notes, and other documents such as project description documents, change request documents, and software design documents. During open coding, abstract conceptual labels were assigned to common data, which capture the characteristics of information interdependencies across modules. The goal of the open coding process was to identify a manageable set of conceptual labels that explain the characteristics of information interdependencies. Thereafter, axial coding was conducted by refining the open codes and exploring the commonalities and variations across the codes. Whenever there were differences between conceptual labels, relevant transcriptions, BRS statements, project description documents, change request documents, software design documents, and observation notes were re-analyzed to understand the reasons for the differences. The knowledge gained through observations was also helpful in understanding the reasons for the differences. After the axial coding, selective coding was performed to derive explanations for the phenomenon. When explanations were lacking, relevant transcriptions, BRS statements, project description documents, change request documents, software design documents, and observation notes were re-analyzed to develop emergent theoretical explanations [76]. See Appendix G for sample codes inducted.

Second, considering well-established knowledge on control theory, a deductive approach [77] was adopted to identify control portfolios employed in projects. Characteristics and sample codes for outcome, behavior, clan, and self-control modes were identified through well-established control theory literature, such as Choudhury and Sabherwal [19] and Kirsch, Sambamurthy, Dong-Gil, and Purvis [78]. The established characteristics and sample codes guided the analysis of control portfolios in projects. Characteristics and sample codes of formal *outcome* and formal *behavior* controls were adapted from Kirsch [14] and Choudhury and Sabherwal [19]. For example, Choudhury and Sabherwal [19] note that one can identify "mechanisms to explicitly specify desired outcomes that were assessed later" (p. 301) as outcome controls. The BRSs employed in this study specified the expected outcomes in detail and led us to the identification of formal outcome controls. Using Choudhury and Sabherwal [19], we identified behavior controls using the "mechanisms that specify appropriate behaviors" (p. 293). Thus, employing in-depth interviews, we identified formal behavior controls used in the cases [19]. We adopted the coding guideline for informal clan controls from Kirsch [14]. There, mechanisms that embrace shared goals and values, such as team members traveling to each other's sites, getting to know each other in informal gatherings, and members sharing pictures are commonly used examples of clan controls [14]. Thus, in this research, characteristics such as team gatherings and team members having lunch together were considered indications of high levels of clan controls. Next, we adapted the coding guidelines for informal self-controls from Choudhury and Sabherwal [19]. For example, Choudhury and Sabherwal [19] identified self-control mechanisms as follows: (1) mechanisms that can encourage or motivate team members to exercise greater self-control and(2) mechanisms that help enhance team members' ability to exercise control. Therefore, when preparing the coding guideline, we considered "mechanisms encouraged and motivated team members to make their own decisions and solutions" as one characteristic of informal self-controls. Following the coding guideline (see Appendix H), we categorized the level of formal outcome, formal behavior, clan, and self-controls in each project as main (dominantly used) or auxiliary (rarely used). See Appendix I for sample codes of deduction.

The project performance (i.e., stable/unstable or average) was also identified using a deductive approach by searching for characteristics and sample codes such as stable, success, average. The deductive analysis was performed preliminarily on interview transcriptions, and was supported by BRS statements, project description documents, change request documents, software design documents, and observation notes. The knowledge gained through observations was also helpful for identifying the control modes and project performance. See Appendix I for sample codes of deduction. Thereafter, we conducted our next level of analysis to identify potential linkages between characteristics of information interdependencies, controls, and project performance. We moved back and forth between the interview transcripts, BRSs, other supportive documents, observation notes, and the literature to remain true to the extant theory while interpreting the data for emerging concepts [69,79].

5. Initial results—emerging patterns and characteristics of information interdependencies and control portfolios

As explained earlier, we commenced the analysis by inductively analyzing BRSs, interview transcripts, project description documents, change request documents, software design documents, and observation notes to identify the nature of information interdependencies. Each BRS contained information about the constituting modules, including the scope of the modules, the flow of information between the modules, and how different modules were related. For example, the interdependency of information flows between *fund processing* and *trade processing* modules of Project A is outlined in the following manner. The trade processing BRS states,

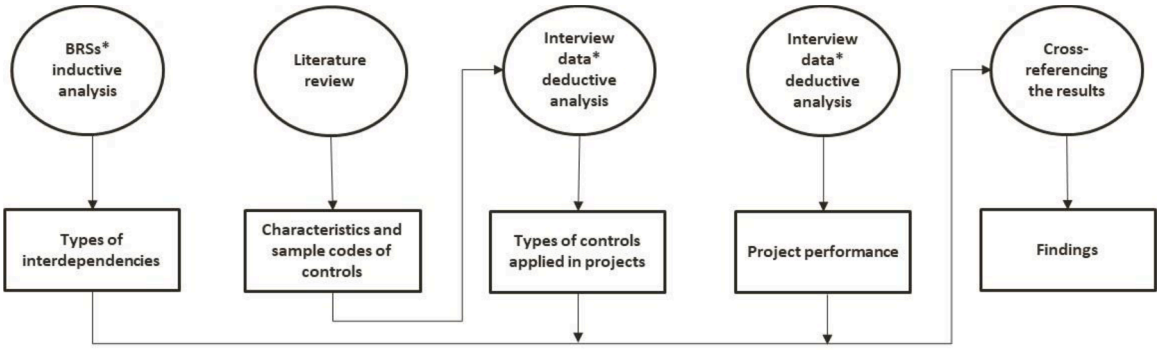


Fig. 2. Data analysis process.

> The Trade Processing business specification outlines the requirements and automated procedures pertaining to processing of trades within the [Company StockEX]. Lifecycle of a trade once entered into the [project A] is defined under the following categories: a) trade entry, b) trade amendments and splits, c) trade confirmations and rejections, d) contract and bill generation and e) printing / dispatch of contract notes. Business requirements for the above procedures are covered in this volume.

The fund processing BRS of Project A mentions that

> Following accounts at a minimum may be created for transactions captured for the current process of trade processing: client cash settlement—the ledger account where fund settlement related client transactions will be captured, e.g. buy/sell trades, cash receipts from client, payments to client, cash withdrawals and etc.

The above statements indicate that the fund processing module manages the fund settlement activities related to the trades that are entered through the trade processing module. The analyses of such information flows yielded two main observations on (i) the nature of information interdependencies occurs modules and (ii) the nature of control portfolios which are being employed to manage such interdependent modules.

5.1. The nature of information interdependencies between modules

Our in-depth analyses of the information flows of the BRSs (i.e., modules) highlighted that some modules mainly had information outflows and did not receive much information from other modules. Such modules demonstrated characteristics of a dominant and independent nature. The analysis revealed that the dominant modules ultimately created information interdependencies with the submissive modules. For example, analysis of trade processing, stock processing, and fund processing BRSs indicated that the trade processing module is a dominant module, as it creates the foundation for submissive modules such as fund processing and stock processing. In other words, the *trade processing* module—which was recognized as the dominant module—created information interdependencies with submissive modules such as (1) the *general accounting and journal entries module*, which calculates account balances, (2) the *fund processing module*, which updates relevant fund accounts, and (3) the *stock processing module*, which manages stock-related transactions.

The information interdependencies between the submissive *fund processing module* and the dominant *trade processing module* can be observed in the following manner. The fund processing BRS of Project A mentions that "Following accounts at a minimum may be created for transactions captured [under] trade processing [module]." In turn, the information interdependencies between *trade processing module* and the submissive *stock processing module* in calculating stock-related delivery obligations of the trades captured under the dominant *trade processing module* can be highlighted through the following quotation. The stock processing BRS of Project A mentions that "Upon initiation of the trade processing process, accounting entries will be posted to the relevant accounts (trades which successfully pass validations will be processed further resulting in the calculation of delivery obligations and updates to the relevant delivery accounts)."

Associated with the dominant and submissive modules of ISDs, the analysis of BRSs identified two patterns of information interdependencies between modules: (i) unidirectional—the information flows only from a dominant module to a submissive module, and (ii) bidirectional—the information flows both ways between dominant and submissive modules. For example, we discovered repeated patterns where information would only flow out from the trade processing module to other modules within a project. For example, Project E, which focused on developing a post-trade solution for the clearing and settlement of the executed trades, showed the dominant role of the trade processing module in relation to other modules such as (a) the clearing and settlement module and (b) the cash management module. The BRS of the "clearing and settlement module" of project E mentions that "All trades must be settled against a specific investor account. The investor account information will not be entered at the time of the trade but will have to be allocated by the broker as part of the clearing process."

Thus, all transactions entered through the trade processing module are settled through the clearing and settlement module, indicating that information flows from the trade processing module to the clearing and settlement module, but not vice versa. On the other hand, the trade processing module and the trade validation and enrichment module of Project B received information from each other, making them an example of bidirectionality. The trade processing BRS specified that "If a trade passes the initial basic validations (as discussed in the trade validations and enrichment specification) it will be considered to be in a pending (initial validation) status." The trade validation and enrichment BRS mentioned that "Once a trade is captured, prior to accepting the trade (and posting to trading/settlement accounts), the following activities need to take place."

Moreover, in some projects, information flows between modules were established only when required, in an ad hoc fashion. For example, Project D involved the development of software solutions to detect irregular trading behavior. The relationship between the trade processing and submissive modules was established only when it was needed to detect irregular trading behavior. On the other hand, in some of the projects there were *permanent information interfaces* between modules. For example, in project E, the BRS for clearing and settlement mentions the following, making permanent associations for clearing, settlement, and cash management transactions with the trade processing module: "The clearing system will receive trades in all forms of instruments executed on the exchange or regulated public market. All trades must be settled against a specific investor account."

5.2. A portfolio of controls in projects to manage performance

The analyses observed varying control modes used in the modules to manage the project performance. The control portfolios and project performance were mainly identified through the analysis of interview transcriptions, supported by BRSs and observation notes. The insights gained through observations were also helpful in deriving control portfolios and the project performance. Thereafter, we compared the characteristics of inter-modular requirements information interdependencies and control modes of each individual project with those of other projects to identify the linkages between characteristics of information interdependencies, controls, and project performance. Yet we do not assert a causal relationship between project performance and information interdependency. Instead, we simply observed that the nuanced type of information interdependency may align well with a particular type of control portfolio.

When projects were observed, the analyses found evidence of simultaneous execution of multiple control modes. Based on the coding guidelines (see Appendix H), we first categorized the level of formal outcome, formal behavior, clan, and self-control in each project as main (dominantly used) or auxiliary (rarely used). Per Choudhury and Sabherwal [19], outcome control was identified by the mechanisms that specified the desired outcomes of each project. Behavior control was established through the mechanisms that specified the rules, procedures, and processes to be followed in achieving the expected outcomes [19]. For example, all projects employed a detailed BRS to explain the expected outcomes and behaviors (i.e., the rules, procedures, and processes required to achieve the outcomes) for each module. Team members in project E did not have to request changes in BRSs, as the BRSs of the project provided a sufficient level of information about the expected outcomes. Respondent 13 from project E explained; "We didn't have to go and change requirement and so on." This indicates that formal-outcome controls were mainly used in project E. In contrast,

team members in project C struggled to understand the expected outcomes of the project, as the BRSs did not contain a sufficient level of information. Respondent 07 from project C mentioned that "Most of the time, the spec [BRSs] carry out only high-level requirements. If you take [BRSs in] another project very, very detailed than [our project]." Thus, we categorized the level of formal-outcome controls in project C as auxiliary.

Intermediate supervisors in project E conducted daily project reviews to ensure that the project was being executed per the project plan. Respondent 13 from project E discussed that "There are lots of people managing [the project]. There is a layer [on] top of us to manage it. So … our immediate supervisors track each and every percentage of work done … and what is not done and maybe they are the people who report to PMs [project managers]." This indicated that formal-behavior controls were mainly used in project E. However, project plans of project F were frequently updated. Respondent 15 from project F discussed how "We have the changes happening in the plan twice a week because we have the new changes coming into the plan." Thus, we categorized the level of formal-behavior controls in project F as auxiliary.

Similarly, most projects included some measures of clan control. For example, since the team members in project B had high team spirit, they were able to deal with unplanned situations. Respondent 05 from project B mentioned that "Actually, that was the main reason why this project went live. Because, everything didn't happen in the proper, official, standard or expected way. We are just [dealing with] change all the time. But, people had the team spirit that is why the project plan went live." Team members in project D also worked together very cohesively in an informal manner. Respondent 10 from project D mentioned that "It is like [a] family." Thus, we identified that project B and E mainly used clan controls in their projects. In contrast, respondent 03 from project A mentioned that "Ultimately, everybody came up with reasons to cover themselves. When I compare it to the other projects I worked on, teamwork was low." Thus, we categorized the level of clan controls in project A as auxiliary.

6. Cross-case analysis results

During cross-case analysis, the nature of information interdependencies and the nature of control modes for each individual project were grouped and compared with those for other projects. We used perspectives from control theory and the literature on information interdependencies as sensitizing devices [69], which assisted us in summarizing the characteristics of information interdependencies and control modes.

Based on the nature of information flows between modules, we identified (i) unidirectional and (ii) bidirectional information interdependencies. Moreover, the temporality of the information interdependencies yielded (i) permanent and (ii) temporary information interdependencies. Next, we determined the control portfolio employed in each project. Therein, per Tiwana [18], we observed that the presence of one control mode could minimize the need for other types of control modes. We also established the performance of each project. Subsequently, we commenced our next level of analysis with the intention of identifying potential linkages [80] between characteristics of information interdependencies, and controls. By doing so, we were able to identify (1) four types of information interdependencies, (2) control portfolios used to govern these information interdependencies, and (3) the observed project performance.

6.1. Unidirectional permanent information interdependency

The interdependency between two modules is considered a *unidirectional permanent information interdependency* when there are unidirectional, permanent links between the dominant module and submissive modules (see Fig. 3). While the submissive modules receive information from the dominant module, the dominant module is not dependent on the information received from the submissive modules. The scope and the interfaces of the interdependency are known. In unidirectional permanent interdependency, the formal control mode is established without negotiation between the modules and is the preferred mode to manage them. There are also mechanisms to encourage and motivate the team members to exercise greater self-control. The role of clan control is less in projects where unidirectional permanent information interdependencies are observed.

Fig. 3. Unidirectional permanent information interdependency.

Unidirectional permanent information interdependency was observed in project E, which was focused on developing a post-trade solution for the clearing and settlement of executed trades. The solution provided the ability to execute manually entered trades as well as market trades using trade uploads. In the process of clearing and settlement, the trade owners and relevant accounts were updated. Unidirectional permanent information interdependency was observed between the trade processing, clearing and settlement, and cash management modules. The trade processing module consisted of information required to process a trade, whereas the cash management module contained details on managing cash accounts, calculating the fees and other trade taxations. The cash management BRS specifies the following: "Similar to shares, cash accounts are also maintained for each transaction [i.e., each trade] in the system." The clearing and settlement module explained the processes for managing risk between a trade taking place and being settled, along with the exchange of cash and assets between buyers and sellers following a trade. Moreover, it was identified that there were no updates to the trade processing module based on the feedback of cash management and clearing and settlement modules. Therefore, the direction of the interdependency can be identified as "unidirectional" (i.e., directed from the trade processing module to clearing, settlement, and cash management modules, not vice versa). Throughout the project, the submissive modules were executed per the requirements of the trade processing module (i.e., the dominant module). Thus, the relationship between the trade processing module and the submissive modules was "permanent." The scope and interfaces of the modules were known by the team members. For example, the clearing and settlement BRSs of project E mention that "The purpose of the clearing system is to get pre-matched trades ready for settlement, to calculate settlement obligations and, when required, to apply these settlement obligations," indicating that all trades processed through trade processing module were settled through clearing and settlement module.

BRSs explained the expected outcomes and behaviors (i.e., required rules, procedures, and processes to achieve the outcomes) of each module. There were minimum updates to requirements specified in BRSs of project E. Because the team members completed their tasks mainly based on BRSs, outcome and behavior controls were identified as "dominant" controls. Since the submissive modules simply use information from the dominant module, formal controls were established without negotiation between the modules. For example, respondent 13 from project E explained, "We didn't have to go and change requirements and so on." Respondent 13 continued, "For only few changes, we [software engineers] have to go and ask [the business analysts], can we make this change?" Team members of the project mainly received isolated work. According to Respondent 13 from project E, "Most of the time they [team members get] isolated work." The first author observed that the team members do not have much communication with each other. Rather than communicating with each other, team members were focused on completing tasks assigned to them individually. Since the team members rarely collaborated, clan controls

were identified as "auxiliary" controls. There were mechanisms to encourage and motivate the team members to exercise greater self-control. For example, business analysts could independently decide the best methods for delivering client requirements. Respondent 13 mentioned that "Most of the time they [business analysts] would say yeah, this is the best way to do it and they would come to a conclusion." Therefore, self-control was considered a "dominant" control. Based on the respondent's feedback, the project performance was identified as "above average." For example, respondent 13 mentioned that "There were many issues; however, it is up to an accepted level currently. It has bugs. I think it [the project] is little bit above average."

6.2. Unidirectional temporary information interdependency

The interdependency between two modules is considered a *unidirectional temporary information interdependency* when there are unidirectional (i.e., the information would flow only from one module to the other, but not vice versa) temporary relationships (i.e., the connections between modules are established only when it is required) between the dominant and submissive modules (see Fig. 4). In a unidirectional temporary information interdependency, the scope and interfaces of the interdependency are known. Thus, formal controls are established for the known interactions, while clan control and self-controls are also used to govern the modules.

Unidirectional temporary information interdependency was observed in project D, which focused on developing a software solution for detecting irregular trading behavior. The project description document of project D mentioned that "Manipulative trading behavior such as front running, insider trading, wash sales, layering the book, and marking the close are detected." Since the solution was focused on detecting irregular trading behavior, the submissive modules of the projects (e.g., business intelligence module and the alert and case management module) were executed per the outputs of the trade processing module. Therefore, the link between the trade processing module (considered as the dominant module) and the business intelligence module and the alert and case management module (i.e., submissive modules) was "unidirectional" (i.e., the submissive modules are executed per the instructions generated from the trade processing module, not vice versa). A relationship between the dominant module and the submissive modules was established only when it was needed to detect irregular trading behavior. Therefore, the relationship was a "temporary" connection between the dominant module and the submissive modules. The scopes of the dominant module and the submissive modules were known to the team members. While the scope of the trade processing module was to execute the trades, other modules were focused on detecting irregular trade behavior. Therefore, team members were able to take full responsibility for modules assigned to them.

Because the BRSs consisted of detailed information about expected outcomes and behaviors, team members completed their tasks primarily based on BRSs. The outcome and behavior controls were established for the known interactions between modules. Since team members completed their tasks mainly based on BRSs, the outcome and behavior controls were identified as "dominant" controls. The team members in project D worked together as a family. For example, respondent 10 from project D mentioned that "It is like [a] family." The first author observed that the team members were talking to each other in a friendly manner and having informal chats with each other. Therefore, clan controls were identified as a "dominant" control. When there were last-minute time overruns, team members worked extra hours. Moreover, there were mechanisms to encourage and motivate the team members to exercise a greater level of self-control. Thus, self-control was considered as a "dominant" control. Based on the respondent's feedback, the performance of project D was identified as "stable." For example, respondent 12 from project D mentioned, "It is sort of stable. You can't say that there are not any issues. With each and every new thing [i.e., components] that is getting added, always introduce issues. And that component might not be like the rest of the system. But as a whole, it is stable product."

6.3. Bidirectional permanent information interdependency

The interdependency between two modules is considered a *bidirectional permanent information interdependency* when there are permanent links between the two modules and when each module depends partially on the other for deriving information (see Fig. 5). The scope of the interdependency is pre-established; however, the interfaces through which they interact are loosely defined. Therefore, the use of formal control modes is largely ineffective. Because the interfaces are not well established, the interfaces are managed through clan control. The BRSs do not have detailed information; thus, team members are able to elaborate requirements, leading to self-control.

Bidirectional permanent information interdependency was observed in projects B, C, and F. Project B developed a real-time clearing system that manages post-trade activities. In project B, the trade validation and enrichment, market trade input and trade details, and trade processing BRSs had permanent information interdependencies with each other. The trade validation and enrichment module consisted of information on (1) trade validation to ensure that all required information (for the activities of clearing and settlement) was present in a trade, and (2) trade enrichment to include any missing information such as settlement details and settlement dates. The market trade input and trade details module consisted of information on capturing trades from various markets and converting the received trades to internal formats to maintain all information required to manage the trade through the life cycle of clearing and settlement. The trade processing module consisted of information on viewing trade information, the trade update process, and trade allocation. The trade validation and enrichment BRS mentioned that "Once a trade is captured, prior to accepting the trade (and posting to trading/settlement accounts), the following activities need to take place. This specification will cover the validation and enrichment activities. Upon completion of these activities the trade will be deemed to have been accepted into the system." The trade processing BRS specified that "If a trade passes the initial basic validations (as discussed in the trade validations and enrichment specification) it will be considered to be in a pending (initial validation) status." This indicates that these BRSs are dependent on each other for deriving information and execution information throughout the project execution. Therefore, it can be concluded that there is a "bidirectional" (i.e., BRSs are dependent on each other for deriving information) "permanent" interdependency (i.e., the connection remains throughout the project). Since projects C and F focused on developing platforms to integrate modules, it was necessary to update the dominant module (i.e., trade processing module) per the requirements of the submissive modules as well. For example, respondent 07 from project C stated that "[We] ask our client first whether they are okay with that, then we accept it, and change the [trade processing] spec [BRS] as well." Thus, the connection

Fig. 4. Unidirectional temporary information interdependency.

Fig. 5. Bidirectional permanent information interdependency.

between the dominant module and the submissive modules in projects B, C, and F can be considered "bidirectional." Since the connections between dominant and submissive modules were maintained throughout the project, the information interdependencies between modules can be considered "permanent."

The team members of projects B, C, and F had a proper understanding of the scope of modules. For example, the statistics BRS of project F mentions that "The provision of such data is considered to be out of scope for the trading platform." Market trade input and trade details BRS of project B clearly stated the scope of the module: "These statistics will account for trades received only via the direct market source. It will not consider trades reported from … trading or manually entered/uploaded by an administrative user." However, when team members wanted to integrate a new module with existing modules, they had to develop interfaces between the modules. Respondent 09 from project C stated, that "We need to get the interface done and then test the requirement and the functionality." Thus, it can be concluded that the scope of the modules in projects B, C, and F was pre-established; however, interfaces through which modules interact were loosely defined.

The BRSs of projects B, C, and F did not include sufficient information about the client requirements. For example, respondent 07 from project C discussed how "Most of the time, the spec [BRS] carries out only high-level requirements." Respondent 08 from project C mentioned that "Talking about [our project], the specs [BRSs] are little bit loose. So, it is little bit tough to do development based mainly on those specs." Moreover, project plans were updated frequently. For example, respondent 15 from project F mentioned that "We have changes happening in the plan twice a week because … we have new changes coming into the plan." As a result, it was difficult to establish outcome and behavior controls. Since outcome and behavior controls were not well established, the module interfaces were established through clan control. Per respondent 08 from project C, "We have to interact with the BAs [business analysts] a lot and get clarification." Respondent 15 from project F mentioned that "First we engage with the team and find the solution." The first author observed constant communications between team members assigned to different modules. Although they mainly discussed the items specified on BRSs, they also had some informal conversations as well. The team members had lunch together at the same time, which indicated high levels of clan control. Therefore, clan control was identified as a "dominant control," whereas outcome and behavior controls were identified as "auxiliary controls." Since the BRSs did not include detailed information, team members of project B, C, and F were able to elaborate requirements. For example, respondent 07 from project C mentioned that "They [team members] have the freedom there, as the spec is at high level. So, they actually get quite a freedom there." Thus, self-control was identified as a "dominant" control.

While the performance of project B was "average," the performance of projects C, D, and H was identified as "stable." For example, respondent 06 from project B mentioned that "Total solution it is in average status," respondent 09 from project C mentioned "Pretty much stable," and respondent 17 from project F mentioned that "compared to other projects, this project is pretty stable."

6.4. Bidirectional dynamic information interdependency

The interdependency between two modules is considered a *bidirectional dynamic information interdependency* when there are undefined links between the two modules and each module depends partially or fully on the other for deriving information (see Fig. 6). Bidirectional dynamic information interdependencies are observed in projects at their early stages, especially when agreeing with specifications and deliverables. It is naturally hard to employ formal controls, as the evolution of specifications makes it impossible to develop concrete targets. It is difficult to establish clan control in projects with bidirectional dynamic information interdependency. This is because the evolution of specifications creates unmanageable targets for team members, which ultimately minimizes team spirit. Because the functionalities are not well described, team members have more self-control, where they can suggest and develop alternative functionalities.

Fig. 6. Bidirectional dynamic information interdependency.

Bidirectional dynamic information interdependency was observed in project A, where there were no formal links between modules, yet modules depended partially or fully on other modules. Information interdependencies were observed between the trade processing, fund processing, stock processing, and brokerages, taxes, and charges modules. When a trade was processed, the relevant fund accounts were updated by the "fund processing" module, whereas the relevant stock accounts were updated by the "stock processing" module. The brokerages, taxes, and charges module was responsible for calculating brokerages, taxes, and charges per the trade processing module. The dominant module (i.e., the trade processing module) had to be updated per the requirements of the submissive modules (such as brokerages, taxes, and charges). For example, the trade processing BRS was updated including "strategy order ID," which was needed to apply the correct brokerage scheme specified in the brokerage taxes and charges BRS. The trade processing BRS was updated by including the following statement: "The Strategy Order ID will be used only as an indicator to identify the strategy orders separately and to apply the correct brokerage scheme. Please refer to Volume 06, the Brokerages Taxes and Charges BRS for this functionality." Moreover, as specified in the revision history of the fund processing BRS, during the project execution, the accounting structure of project A had to be removed, as the structure could not be implemented without implementing another interdependent module. Respondent 01 from project A mentioned that "We had to remove the functionality completely." Since the trade processing BRS had information interdependencies with the fund processing BRS, it was necessary to update the trade processing BRS as well. The fund process BRS mentions that "The whole account postings section was removed since the same had been explained in the Trade Processing spec." This indicates that the dominant module (i.e., the trade processing module) and the submissive module (i.e., the fund processing module) rely partially or fully on each other for deriving information (i.e., "bidirectional") and the information interdependencies are not clearly identified (i.e., "undefined").

Since the BRSs were frequently updated, it was difficult to establish outcome and behavior controls. For example, respondent 02 mentioned that "They [software engineers] have to change certain things, because the document [BRS] is changing continuously." Respondent 01 from project A mentioned that "So, after that, what we [business analysts] did was, we just updated the BRSs and sent another version." Respondent 01 continued, "They [clients] basically signed off on the business functionality. So, the BRS didn't specify exactly how we are going to give [the software solution] to you." The evolution of specifications created unmanageable targets for the team members, leading to a minimum level of team spirit. According to respondent 03 from project A, "Ultimately, everybody came up with reasons to cover themselves. When I compare it to the other projects I worked on, teamwork was low." Respondent 01 mentioned that "Most of the time, when we [consultants] suggest a problem or suggest a solution [software engineers mention that] we can't do this." Respondent 01 continued, "Team spirit was really lower [than other projects]." The first author observed tensions between team members: Some team members were not that friendly with the others. Therefore, outcome, behavior, and clan were identified as "auxiliary controls." Since the functionalities were not well

elaborated, team members had more self-control, so that they were able to suggest and develop alternative functionalities. For example, respondent 01 from project A mentioned that "We have to come up with the solutions and we have to elaborate on the functionalities and one other thing I want to emphasis is when we started the project, we didn't know anything about their operations. For example, you can have a functionality, but you don't know how they currently operate, then the scope of the functionality becomes very open. Since we don't know the exact way that they are doing that functionality, we just try to come up with several alternatives which were never used by the client." Thus, self-control was identified as "dominant control." Per the respondent's feedback, performance of project A was identified as "unstable." For example, respondent 02 mentioned that "It is almost at the exit procedure level. So, we were [in] discussion how we get out from the project, because it has dragged for so long, three and half years."

Appendix J illustrates the use of various types of data (e.g., interview transcripts, BRSs, and observation details) in the analysis process and the causal links between interdependency, control, and project performance using the "bidirectional dynamic information interdependency" type as the example. We followed a similar approach in deriving other three types of information interdependencies (i.e., unidirectional permanent information interdependency, unidirectional temporary information interdependency, and bidirectional permanent information interdependency), control portfolios, and performance of other projects.

Fig. 7 graphically presents the four types of information interdependencies, applied control modes, and the observed project performance.

The project with unidirectional temporary information interdependency (project D) considered all four types of control modes as main control modes and they were able to execute and maintain project in stable condition. This highlights that projects with unidirectional temporary information dependency can be managed effectively by using all four types of control modes as main control modes depending on the project situation. Because the project with bidirectional dynamic information interdependency (project A) considered outcome, behavior, and clan control modes as auxiliary controls, the project was in an unstable condition. This indicates that projects with bidirectional dynamic dependency cannot be managed effectively by considering using outcome, behavior, and clan control modes as auxiliary controls. While projects with bidirectional permanent dependency (projects B, C, and F) considered outcome and behavior controls as auxiliary controls, clan controls and self-controls were considered as main control modes. Because these projects were in average or stable condition, it can be concluded that projects with bidirectional permanent dependency can be managed by considering outcome and behavior control as auxiliary controls while considering clan controls and self-controls as main controls. Further, the project with unidirectional permanent dependency (project E) considered outcome, behavior, and self-control modes as main controls; the project was in above average condition. This indicates that projects with unidirectional permanent dependency that were at "above average" levels of performance considered outcome, behavior, and self-control modes as main control modes.

7. Implications for research

Using Fig. 7 as a summary of our study, we identify several significant implications for research.

First, we recognize the complexity of modular developmental environments of ISD organizations. Studies have shown that ISD organizations often engage in remote modular developmental environments [81]. While modularity has been hailed as a "tried-and-tested" approach to developmental tasks, our study demonstrated the challenges associated with the flow of information, modularity, and the control portfolio employed in managing a project.

Second, our study identified how the dominant and submissive modules can usefully be determined for future research. Observing the flow of information and the nature of information interdependencies, the study identified directionality (unidirectional vs. bidirectional) and temporality (permanent vs. temporary) could be employed to define a module. While past research has employed several useful classifications such as the type of workflow [82], type of routines [8,83], project attributes [84], and type of goals and level of resources [85], no other past study had employed the nature of information flow to classify a module.

Third, this is one of the few studies to highlight the importance of considering information interdependencies when governing ISD projects. Our findings add further substance to the claims presented in Tiwana [2], who explained how minimizing interdependencies between outsourced systems and other software applications is a substitute for control mechanisms. Kirsch [86] pointed out that ISD projects may include task interdependencies, leading to changes in the choice of control modes. According to Ouchi and Maguire [15], managers prefer outcome controls when tasks are interdependent. Although previous research has explained how to develop control portfolios considering the level of interdependencies (e.g. high or low; [2,68]), there is a lack of research that explains how to develop control portfolios based on the *types of information interdependencies.* Moreover, previous research does not sufficiently explain which control mechanism can be dominant—"main" and "auxiliary"—depending on the types of information

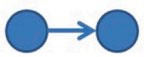
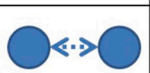
Information Interdependency				Control Portfolio				Project Performance
Information Interdependency Types and Characteristics			Graphical Representation	Formal-outcome control	Formal-behaviour control	Informal-clan control	Informal self-control	
Unidirectional Permanent Information Dependency	Direction of information flows	Uni-directed		Main	Main	Auxiliary	Main	E – Above average
	Temporality	Permanent						
Unidirectional Temporary Information Dependency	Direction of information flows	Uni-directed		Main	Main	Main	Main	D – Stable
	Temporality	Temporary						
Bidirectional Permanent Information Dependency	Direction of information flows	Bi-directed		Auxiliary	Auxiliary	Main	Main	B – Average C - Stable F – Stable
	Temporality	Permanent						
Bidirectional Dynamic Information Dependency	Direction of information flows	Bi-directed		Auxiliary	Auxiliary	Auxiliary	Main	A – Unstable
	Temporality	Dynamic Undefined						

Fig. 7. Results—types of information interdependencies, relevant control portfolios, and project performance.

interdependencies. For example, observing the results of Fig. 7, we note that projects with bidirectional permanent information interdependency tend to align well with informal types of controls. We extend prior control theory research [2,14,19,68] by highlighting the importance of selecting appropriate control portfolios for various types of information interdependencies.

Fourth, our research highlights the dynamic existence of multiple control modes in a single ISD project. In general, studies assume the existence of a single control mode within a project. For example, Tiwana [2] focused solely on formal controls and did not explicitly discuss the relationship between modularization and clan control. Although the control literature has discussed clan control [62,65], there is a lack of research that discusses the management of multiple clans within a single ISD project. The results from our study indicate that ISD projects consist of several teams with different team goals and objectives, thereby creating multiple clans within a single project. Moreover, these teams are assigned to different modules. Therefore, when projects include information interdependencies, project managers should ensure that clan control is appropriately implemented within intra-project teams (e.g., within the business analyst team and within the software engineering team) and between intra-project teams (e.g., between the business analyst team and the software engineering team) assigned to different requirements modules. Although modularity minimizes the need for organizational control, control is needed to ensure that teams assigned to different requirements modules have a sufficient level of interaction to perform their tasks.

Fifth, our study develops four types of information interdependencies and thus extends the extant research on interdependencies and modularity. Although previous research has discussed types of requirements interdependencies [12,36,37] and highlighted the importance of considering the strength [87] and order [88] of requirements interdependencies, research that identifies requirement information interdependency types based on the flow of information and temporality is scarce. The new classification of information interdependencies that we delineated in our study provides a nuanced view of the interdependencies. Moreover, we explain the innate characteristics of all four types of information interdependencies. Such knowledge will enable future research to derive better insights through a deeper understanding of the nature and complexity of interdependencies in contemporary ISD projects.

Finally, while determining project performance ex ante is elusive at best in every project, we have made an attempt to identify a diagnostic framework (summarized in Fig. 7) that allows researchers to draw insights into the nature of information interdependencies in ISD modules, which could be used determine the most suitable type of control portfolio for an ISD project. While there has been much discussion in the literature regarding ISD project performance and ISD success determinants (e.g., [89–91]), no past study has provided such an approach.

8. Implications for practice

The summarized findings of Fig. 7 have substantial implications for practice as well.

First, the study demonstrates the impact of the modularization of requirements, which is commonly employed to better govern ISD projects. More specifically, the study findings highlight the impact that information interdependencies have on project controls. Thus, it demonstrates that better communication is required between business analysts, software engineers, and project managers when specifying requirements modules—a task traditionally done only by business analysts.

Second, ISD organizations commonly use modular development environments. Our study highlights the challenges associated with the flow of information, modularity, and the control portfolios employed in such environments. Thus, ISD team members working in modular projects could improve their understanding and take necessary steps to manage those challenges.

Third, practitioners can benefit from our findings regarding the various control portfolios that can be employed for different projects. More specifically, the study demonstrates the vital role that BRSs play in project control. If developed with clear instructions, considering all the anticipated information interdependencies, BRSs can serve as a structured formal control mechanism for all requirements modules. However, given the inevitable information interdependencies between modules, managers must employ a portfolio of controls to govern projects.

Fourth, the study provides four information interdependencies that practitioners can use to assess the dominant interdependency type in a project and then develop the appropriate control portfolio to govern the project effectively. The study characterizes the information interdependencies and provides means of how they can be assessed by observing the information flows across modules. Such a structured approach will undoubtedly benefit managers in governing future ISD projects.

9. Limitations and directions for future research

Though the study makes several theoretical and practical contributions, it has a few limitations that need to be acknowledged. Although procedures for internal and external validity were followed, the subjectivity of data collection and analysis may be an issue. In future studies of this nature, generalizability can be improved by using a larger sample of projects representing multiple organizations. This research explained how to develop control portfolios to govern information independencies by using outcome, behavior, clan, and self-control modes. Future research could enrich the findings by integrating recent control theory concepts such as control styles (e.g., enabling/authoritative), control degree (e.g. frequency/intensity), and control types (e.g. social type/procedural; [24,47]) to explain the development of the control portfolio. This research has focused only on developing control portfolios to govern information interdependencies; future research can further explain the development of control portfolios to govern other types of interdependencies, such as interdependencies between projects and organizations. We have used interview data, BRSs, observation notes, and observation information to derive project performance. Future research can use other types of objective data such as hours planned vs. hours expended, budget overruns, schedule overruns, and customer satisfaction scores to derive the performance. A longitudinal study on managing information interdependencies in modularized ISD projects could yield further insights into the phenomenon being observed. For example, sequential and parallel development of requirements modules in ISD projects could be observed through a longitudinal study.

Despite these limitations, our study is one of the few that have examined the nature of information interdependencies within ISD projects and attempted to delineate the rationale for governing projects through appropriate control modes.

10. Conclusions

The objective of this study was to examine how information interdependencies in ISD projects can be governed better using a portfolio of control modes. We examined six ISD projects selected from a single organization that specializes in developing capital market solutions. Our analysis was based on interview data from 17 respondents with different roles in the six projects, company documents such as BRSs, project description documents, change request documents, and software design documents and our own observational notes taken during the data collection process. We grounded our study in control theory to draw upon existing perspectives on project governance using formal and informal control modes. During data analysis, four types of information interdependencies were identified: (i) unidirectional permanent information interdependency; (ii) unidirectional temporary information

interdependency; (iii) bidirectional permanent information interdependency; and (iv) bidirectional dynamic information interdependency. Each of these four "information interdependencies" was discussed and mapped onto the control portfolios and project performance. Based on our analysis, the study demonstrates a clear rationale for using the relevant control modes aligned to specific types of information interdependencies.

CRediT authorship contribution statement

Maduka Subasinghage: Writing – original draft, Formal analysis. **Darshana Sedera:** Writing – review & editing, Supervision, Conceptualization. **Shirish C. Srivastava:** Writing – review & editing, Validation.

Appendix A: Types of interdependencies

Entity of interdependency	Types of interdependency	Sample references
Organizational interdependencies	Pooled interdependencies—employees can perform their tasks without interacting with other employees. Interdependencies occur only because of limited resources. Sequential interdependencies—one employee can commence tasks only when the previous employees have completed their tasks. Reciprocal interdependencies—employees should adjust their behaviors and tasks in order to align with other employees. Goal interdependency—the goals of one group can be achieved only when the goals of other groups have been met. Task interdependency—team members of a group believe that they depend on other groups to be able to complete the assigned tasks. Reward interdependency—the rewards of one group depend on the performance of other groups. Outcome interdependency (includes goal and reward interdependencies) – employees perceive that their outcomes depend on the outcomes of other employees. Means interdependency (includes task, resource and role interdependencies)—employees perceive that they depend on the means of others to achieve expected outcomes. Boundary interdependency—employees perceive that they have continuous interdependencies with other employees.	Ahola [92], Baccarini [93], Ghobadi [94], Ghobadi and D'Ambra [95], Hong [96], Johnson and Johnson [97], Parolia, Jiang, Klein, and Sheu [98], Pee et al. [5], Stea, Foss, and Foss [6], Tan, Tan, Wang, and Sedera [99], Stea et al. [6]; Thompson [100], Wang et al. [84]
Project interdependencies	Knowledge interdependency—the progress of a project depends on the form of information (e.g., domain knowledge, knowledge about past decisions). Process interdependency—a task can be only started after the completion of another task. Resource interdependency—progress of a project depends on resources. Routine interdependency—information interdependencies between repetitive actions conducted by multiple actors.	Cho et al. [7], Dönmez et al. [8], Faraj and Sproull [101], Feldman and Pentland [83], Jiang et al. [85], Lindberg et al. [4], Strode [102]
Structural interdependencies	Structural interdependencies—interdependencies between requirements when they are organized into a structure. Constraint interdependencies—requirements constrain or depend on each other. Cost/value interdependencies—the cost involved in implementing a requirement in relation to the customer's perceived value.	Aurum and Wohlin [12], Carlshamre and Regnell [103], Dahlstedt and Persson [36], Ramesh and Jarke [104]
	Conditions—create restrictions on requirements. Content—content interdependencies in requirements modules. Documents—links between different documents that are required to trace information or find examples of requirements modules. Evolutionary—a particular requirements module has been developed based on another requirements module. Abstraction—abstractions between different requirements modules, such as refinements or generalizations.	Dahlstedt [105], Li et al. [106], Pohl [37]

Appendix B: Overview of existing research on interdependencies, modularization, and project governance in ISD projects

Study	Method	Theory	Focus	Main findings
Tiwana [68]	Quantitative method Data collection in three phases over five years (2009–2013) from 342 independent extension developers	Control theory	The interplay between control and extension modularization	Complementarity between input controls and extension modularization increases the project performance.
Tiwana [2]	Quantitative method Data from 120 software outsourcing alliances	Control theory	To identify whether technological modularity substitutes for control	Process control, outcome control, and modularity independently enhance alliance performance. Modularity and control are imperfect substitutes; modularity minimizes the effect of process control on alliance performance but not of outcome control on performance.
Cataldo [3]	Quantitative method Four-stage study using	Socio-technical congruence	To identify the work dependencies and to investigate the impact of work dependencies on the development process	Defines a method to measure socio-technical congruence (i.e., the relationship between work dependencies and coordination patterns).

(*continued on next page*)

(*continued*)

Study	Method	Theory	Focus	Main findings
	software source code files and modification requests			
Lindberg et al. [4]	Qualitative method Using the available data within the repository	Coordination	To explore how an open-source software project manages unresolved development interdependencies	Explained how development and developer interdependencies are associated with different routines (i.e., sequence of activities).
Cataldo et al. [59]	Quantitative method Data from the first four releases of a large distributed system development project	Socio-technical congruence	To explain the impact of technical and work dependencies on software development productivity	Identification of the accurate set of product dependencies determines the relevant work dependencies, and thus ultimately minimizes the resolution time of software modification requests.
Gomes and Joglekar [107]	Quantitative method 71 tasks carried out in 11 software development projects	Information processing view of product development and transaction cost economics	To identify the effect of task modularity on transactional efficiency and software project completion time	Task modularity increases the transactional efficacy. Increase in modularity decreases the development time and coordination effort.
Conley and Sproull [108]	Quantitative method Data collected from 203 software releases in 46 open-source projects hosted on SourceForge.net	Software modularity	To identify the relationship between software design modularity and software quality	Software modularity reduces the software complexity, increases the number of static software bugs, and has a mixed relationship with percentage of software bugs closed.
Narduzzo and Rossi [109]	Qualitative method Case studies and indirect observations	Software modularity	To identify the relationships between organizational structure and architecture design in modular projects and to explore how the modularity copes with unexpected interdependencies	Increasing the degree of modularity and violating the information hiding principle are important in open source software projects.

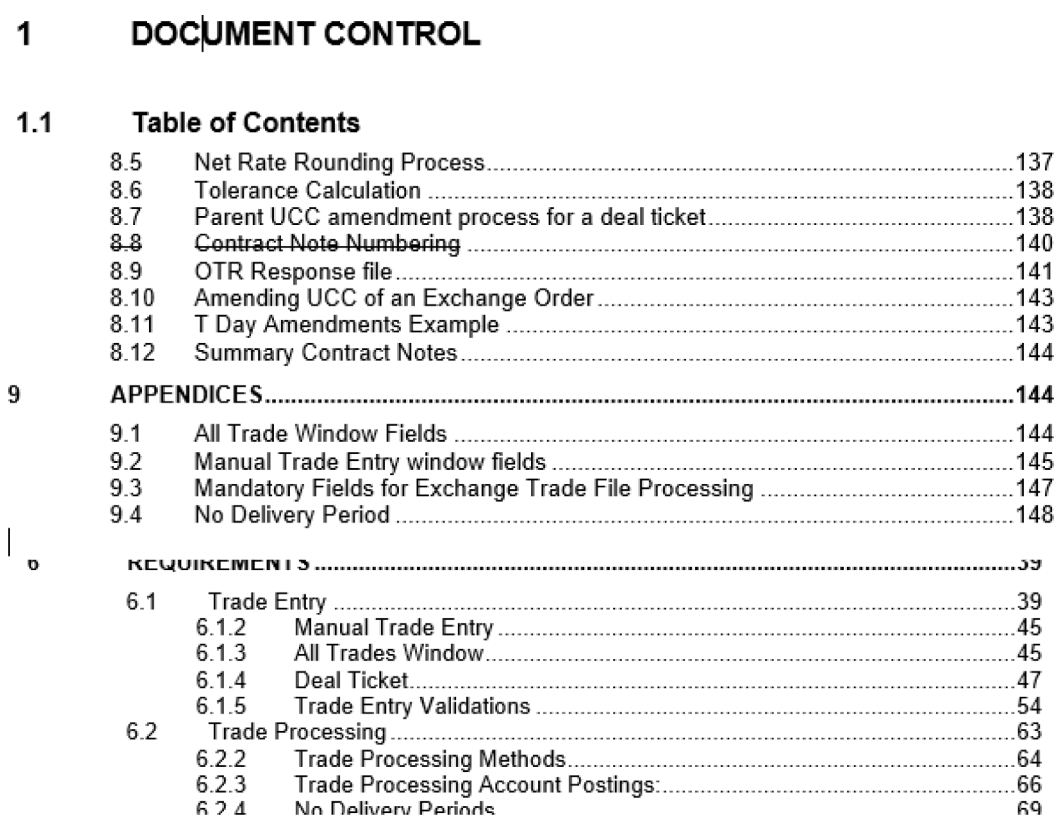
Appendix C: Company description

Company StockEX is a medium-sized ISD company engaging in capital-market-related ISD. It has been in the business for over 10 years, employing over 300 staff. The company specializes in developing software solutions for capital markets, with more than 25 capital market clients all over the world. Those solutions provide the ability to trade using multiple assets such as equities, commodities, derivatives, and debt. The software solutions include functionalities of multiple trading methods such as auctions and continuous matching. Software solutions provide the ability to trade in multiple market structures such as regulated exchange and the over-the-counter markets. Moreover, the company develops post-trade applications, wherein the settlement of trades is automated. Post-trade applications provide the central clearing and settlement for trades.

Company StockEX provides systems integration services for clients in different industry sectors such as the financial and telecommunication industries. Furthermore, the company offers consultancy services and IT infrastructure services. It is involved in developing the clients' requested functionalities of ISD projects even after the projects go live. The company includes several partners all over the world, who provide the required outsourcing services and hardware and software services. Specialized industry teams interact with ISD teams and marketing departments to develop software solutions which are aligned with client expectations. The company tries to be agile and responsive to changes in the global market. All software solutions are developed according to specific client requirements and to facilitate innovation and collaboration with client's business partners.

Software solutions from the company are based on an advanced technology platform, which uses distributed parallel software design. Since the company utilizes a common platform, company products are fully interoperable and can be extended to provide exchange solutions for different market types. Software solutions are developed using a unique software tool of the company, which facilitates developing software for real-time business management. By providing the ability to update business rules, software solutions offer flexibility and competitive advantages to clients.

Appendix D: A sample business requirements specification document – table of contents



1 DOCUMENT CONTROL

1.1 Table of Contents

Appendix E: List of guiding interview questions

1. Can you please describe Project X? To maintain confidentiality, the names of the projects were disguised.
2. What sort of issues do you encounter in your project?
3. Can you describe the documents and software systems that your team uses to transfer the client requirements?
4. To what level do you document the client requirements?
5. Are there any other documents and software systems that your team uses as contracts between the client and you?
6. Can you please describe the penalties, rewards, and time allocations of your project?
7. Can you discuss the issues you face when you are controlling a project?
8. What knowledge is required for your team members to develop products that satisfy client requirements?
9. What knowledge do team members have about the contracts and requirement documents?
10. To what level do you follow the document during day-to-day activities?
11. How do you describe the behavior of your project team members? Are they flexible to provide more information than what is mentioned in the requirement documents?
12. Can you please describe the team spirit and shared values and beliefs of the team?
13. To what extent do you amend the requirement documents, time estimations and project templates according to the requests from the team members?
14. How does your team manage day-to-day operations and quick decisions?

Appendix F: Rigor in research

Rigor in research sought	Description	Case study strategy	How the strategy was applied in this study
Dependability	Two researchers independently arrive at the same conclusions by assessing the same data set.	Provide adequate information about the phenomenon of interest.	Adequate information about the case organization and projects were provided in the research methodology and data collection section and Appendices.
Credibility	Readers accept that the inferences of the research are true.	Provide evidence of data triangulation, data collection techniques, and data analytic procedures.	Data triangulation was conducted using interview data, observations, BRSs, design documents, project descriptions, and data collected from official websites. Data collection and data analysis techniques were explained in the methodology and data analysis sections.

(*continued on next page*)

(*continued*)

Rigor in research sought	Description	Case study strategy	How the strategy was applied in this study
Transferability	Identify whether the findings can be generalized.	Provide in-detail descriptions of the research context, so that the reader can identify to which extent the findings can be generalized.	Detailed description of the research context is provided in the research methodology and data collection section and the Appendices.
Confirmability	To the extent that the results can be confirmed by others.	Preparation of a topic guide. Presenting research findings.	A topic guide was prepared including detailed objectives of the study, pre-understandings, and our influences. Research findings were presented at seminars and received feedback.
Unbiased results	Interview data can be subjective; therefore, obtaining results from one person can generate inaccurate and bias results.	Interview many participants from each project. Use of objective data.	From each project, three participants were interviewed. Results were gained by comparing and contrasting all three members' opinions—this minimized the bias in the results. Objective data (e.g., BRSs and project descriptions) were used to reduce the ambiguity of data and generate unbiased results.

Note: Adapted from Bhattacherjee [110] and Whitman and Woszczynski [111].

Appendix G: Induction: sample codes of information interdependencies

Please note that the sample codes have been explained in the results sections.

Open codes	Axial codes	Selective codes
"The purpose [open code: scope known] of the clearing system is to get pre-matched … trades [open code: unidirectional] ready for settlement, to calculate settlement obligations and, when required, to apply these settlement obligations [open code: scope known]." [Project E, clearing and settlement BRS, p. 6] "The clearing system will receive trades [open code: unidirectional] in all forms of instruments executed on the exchange or regulated public market." [Project E, clearing and settlement BRS, p. 6] "Similar to shares, cash accounts are also maintained for each transaction [open code: permanent] in the system." [Project E, cash management BRS, p. 4] "All trades must be settled against a specific investor account [open code: permanent]. The investor account information will not be entered at the time of the trade but will have to be allocated by the broker as part of the clearing process." [Project E, clearing and settlement BRS, p. 9]	[Direction of information flows in modules] Unidirectional [Temporality of information interdependencies] Permanent [boundaries of modules] The scope and interfaces of the modules were known	Unidirectional Permanent Information interdependency
"Manipulative trading behavior such as front running, insider trading, wash sales, layering the book, and marking the close are readily detected [open code: scope known]." [Project D, project description document, p. 2] "Integrated BI [business intelligence] module to generate reports, and dashboards accessing internal as well as multiple external data sources" [open code: scope known]. [Project D, project description document, p. 6]	[Direction of information flows in modules] Unidirectional [Temporality of information interdependencies] Temporary [boundaries of modules] The scope and interfaces between modules were known	Unidirectional Temporary Information interdependency
"These statistics will account for trades received only via the direct market source. It will not consider trades reported from … trading or manually entered/uploaded by an administrative user [open code: scope pre-established]." [Project B, market trade input and trade details BRS, p. 10] "The provision of such data is considered to be out of scope for the trading platform [open code: scope pre-established]." [Project F, statistics BRS, p. 4] "We need to get the interface done and then test the requirement and the functionality." [open code: interfaces are loosely defined] [Respondent 09, Project C] "Once a trade is captured, prior to accepting the trade [open code: bidirectional, permanent] (and posting to trading/settlement accounts), the following activities need to take place … This specification [validation and enrichment BRS] will cover these validation and enrichment activities [open code: scope of module pre-established]. Upon completion of these activities, the trade will be deemed to have been accepted [by the trade processing BRS] [open code: bidirectional] into the system." [Project B, trade validation and enrichment BRS, p. 5] "Failed and rejected status trades will be referred back to the gateway by communicating with the trading source [open code: bidirectional]." [Project B, market trade input and trade details BRS, p. 10]	[Direction of information flows in modules] Bidirectional [Temporality of information interdependencies] Permanent [boundaries of modules] The scope of the modules was pre-established Interfaces through which modules interact were loosely defined	Bidirectional Permanent Information interdependency
The trade processing BRS was updated, including "strategy order ID," which was required to apply the correct brokerage scheme specified in the brokerages, taxes and charges BRS. This indicated that the foundation module (i.e., the trade processing module) was updated per the requirements of other modules. The trade processing BRS was updated by including the following statement: "The Strategy Order ID will be used only as an indicator to identify the strategy orders separately and to apply the correct brokerage scheme. Please refer to Volume 06, the Brokerages Taxes and Charges BRS for this functionality [open code: bidirectional]." [project A, Trade processing BRS, p. 45] "The whole account postings section was removed since the same had been explained in the Trade Processing spec" [open code: scope of the module was undefined]. [project A, Fund	[Direction of information flows in modules] Bidirectional [Temporality of information interdependencies] Undefined [boundaries of modules] The scope and interfaces between modules were undefined	Bidirectional Dynamic Information interdependency

(*continued on next page*)

(*continued*)

Open codes	Axial codes	Selective codes
processing BRS, p. 5] "We had to remove the functionality completely [open code: scope of the module was undefined]." [Respondent 01, Project A]		

Appendix H: Categorization standards/coding guideline of deductive coding

Formal outcome control: Main
The BRSs specified the expected outcomes in detail
Business analysts made few updates to or did not update the BRSs at all, which indicated stable expected outcomes.
The developed information system solution aligned with the information specified in BRSs
Team members had to strictly focus on the outcomes mentioned on the BRSs
The project had deadlines to ensure that team members achieved the expected outcomes on time.
Formal outcome control: Auxiliary
The BRSs did not properly specify the expected outcome.
The developed information system solution did not completely align with the BRSs.
Business analysts updated the BRSs several times, which indicated unstable expected outcomes.
The team members did not have to strictly focus on the outcomes mentioned on the BRSs
The project did not have deadlines to ensure that team members achieved the expected outcomes on time.
Formal behavior control: Main
The BRSs specified the expected behaviors in detail
Team members had to strictly follow the expected behaviors mentioned on the BRSs
Team members made few or no updates to the software design and software code (i.e., the procedures that team members should follow to achieve the expected outcomes). Lack of updates to the software design and code indicated stable the formal behavior controls.
The project comprised detailed project plans
Project managers did not frequently update project plans
Project managers conducted project reviews and project meetings to ensure that team members executed the project according to the project plan.
Project managers closely monitored team members' progress
Formal behavior control: Auxiliary
The BRSs did not specify the expected behaviors in detail
Team members did not have to strictly follow the expected behaviors mentioned on the BRSs
Team members made several updates to the project plans, software design, and software code
The project did not comprise detailed project plans
Team members did not have regular project reviews and project meetings
Project managers did not closely monitor team members' progress
Informal clan control: Main
The team shared the same values and beliefs
The team displayed high team spirit
Team members had good interactions, relationships, collaborations, and team gatherings
The team members provided suggestions or clarifications to other team members
When some team members required support to complete the assigned tasks, other team members were willing to provide the required support.
Informal clan control: Auxiliary
The team displayed low team spirit
Team members did not interact frequently
Team members did not collaborate often
Team members had weak relationships and there was lack of informal gatherings
Team members did not provide suggestions or clarifications for other team members' tasks
When a team member required support to complete the assigned tasks, other team members were not willing to provide the required support.
Informal self-control: Main
Mechanisms encouraged and motivated team members to make their own decisions and solutions
Team members used their personal time to complete their tasks
Team members had the freedom to innovate and execute their tasks
Informal self-control: Auxiliary
No/few mechanisms encouraged and motivated team members to make their own decisions and solutions
Team members did not use their personal time to complete their tasks
Team members did not have the freedom / lacked the freedom to innovate and execute their tasks

Appendix I: Deduction: sample codes of control modes and project performance

Please note that the sample codes have been explained in the results sections.

Control modes	Sample codes
Formal outcome and behavior controls	"We didn't have to go and change the requirement, [which was written in BRS] [codes: formal controls, without negotiation]." [Respondent 13, Project E] "For only few changes, we [software engineers] have to go and ask [from the business analysts], can we do this change? [codes: formal controls, without negotiation]." [Respondent 13, Project E] "Most of the time, the spec [BRS] carries out only high-level requirements [codes: difficult to establish formal controls]." [Respondent 07, Project C] "Talking about [our project], the specs [BRSs] are little bit loose. So, it is little bit tough to do development based mainly on those specs" [codes: difficult to establish formal controls]." [Respondent 08, Project C] "We have changes happening in the plan twice a week because … we have new changes coming into the plan [codes: difficult to establish formal controls]." [Respondent 15, Project F] "They [software engineers] have to change certain things because the document [BRS] is changing continuously [codes: difficult to establish formal controls]." [Respondent 02, Project A] "So, after that, what we [business analysts] did was, we just updated the BRSs [codes: difficult to establish formal controls] and sent another version." [Respondent 01, Project A] "They [clients] basically signed off on the business functionality. So, the BRS didn't specify exactly how we are going to give [the software solution] to you [codes: difficult to establish formal controls]." [Respondent 01, Project A]
Informal clan controls	"Question- It means the [team] members … think about the other people and work as the team. They help each other. Answer-Yes. It is like [a] family [codes: clan controls]." [Respondent 10, Project D] "First we engage with the team [codes: clan controls] and find the solution." [Respondent 15, Project F] Most of the time, when we [consultants] suggest a problem or suggest a solution, [software engineers mention that] we can't do this [codes: difficult to establish clan controls]." [Respondent 01, Project A] "Team spirit was really low [than other projects] [codes: minimum clan controls]." [Respondent 01, Project A] "Ultimately, everybody came up with reasons to cover themselves. When I compare this to the other project I have worked, the level of teamwork was low [codes: minimum clan controls]." [Respondent 03, Project A]
Informal self-controls	"Most of the time they would say yea, this is the best way to do it and they would come to a conclusion" [codes: self-controls] [Respondent 13, Project E] "They [team members] have the freedom there, as the spec is at high level. So, they actually get quite a freedom there". [codes: self-controls] [Respondent 7, Project C] "We have to come up with the solutions and we have to elaborate on the functionalities and one other thing I want to emphasis is when we started the project, we didn't know anything about their operations. For example, you can have a functionality, but you don't know how they currently operate, then the scope of the functionality becomes very open. Since we don't know the exact way that they are doing that functionality, we just try to come up with several alternatives which were never used by the client [codes: high level of self-controls]. [Respondent 01, Project A]
Project performance	"There were many issues, however it is up to an excepted level currently. It has bugs. I think it is little bit above average." [Respondent 13, Project E] [codes: above average] "It is sort of stable. You can't say that there are no any issues. With each and every new thing [i.e. components] that is getting added, always introduce issues. And that component might not be like the rest of the system. But as a whole, it is a stable product" [codes: stable] [Respondent 12, Project D] "Total solution it is in average status" [codes: average] [Respondent 06, Project B]. "It is going fine" [codes: stable] [Respondent 08, Project C] "Pretty much stable" [codes: stable] [Respondent 09, Project C] "I think we are in a matured stage" [codes: matured / stable] [Respondent 15, Project F]. "There are only few issues found, I think the product is successful" [codes: successful / stable] [Respondent 16, Project F]. "Compared to other projects, this project is pretty stable" [codes: stable] [Respondent 17, Project F] "It is almost at the exit procedure level. So, we were [in] discussion how we get out from the project because it has dragged for so long, three and half years" [codes: unstable]. [Respondent 02, Project A]

Appendix J: Project A—An example of bidirectional dynamic information interdependency, relevant control portfolios and project performance

We observed bidirectional information interdependencies between fund processing and trade processing modules of project A. As specified in the revision history of the fund-processing BRS, the accounting structure of the system had to be removed because the accounting structure could not be implemented due to the information interdependencies between the requirement modules. As a result of removing the accounting structure, the business analysts had to modify 116 specification points, make amendments to 73 points, and clarify a further 29 points of the fund-processing BRS. Removing the accounting structure created the need for more updates in the fund-processing BRS toward the end of the life cycle. For example, on August 13, the fund-processing BRS was updated by (1) adding or deleting 85 specification points, (2) amending 33 specification points, and (3) clarifying a further 31 specification points. Table J.1 demonstrates the number of updates in the fund-processing BRS.

Because the trade-processing BRS had information interdependencies with the fund-processing BRS, it was necessary to update the trade-

Table J.1
Number of updates in the fund-processing BRS.

Date	Number of spec point updates		
	Added/deleted[a]	Amended[b]	Clarified[c]
07/06	6	8	0
14/06	1	6	2
22/06	19	20	8
05/12	116	73	29
26/02	7	3	1
03/03	21	15	0
31/05	26	14	0
13/08	85	33	31
27/02	20	6	0

[a] Added/deleted: A spec point was added to or deleted from the BRS.
[b] Amended: A spec point was amended.
[c] Clarified: A spec point was clarified, including clarification information.

processing BRS as well. The business analysts had to identify the information interdependencies between the two modules, so that they could update both BRSs. This indicates that the information interdependencies between the fund processing module and the trade processing module were not clearly defined. As a result of fund-processing BRS updates, 92 trade-processing specification points were added, 35 points were amended, and a further 26 points were clarified on February 2. TableJ.2 demonstrates the number of updates in the trade-processing BRS.

The fund-processing module relied on the information transferred from the trade-processing module as well. Once the trades are processed through

Table J.2
Number of updates in the trade-processing BRS.

Date	Number of spec point updates		
	Added/deleted	Amended	Clarified
20/09	38	10	0
31/10	19	5	3
02/02	92	35	26
02/03	2	14	0
25/06	13	15	0
25/07	37	37	0

the trade processing module, the relevant account entries are managed through the fund-processing module. Fig. J.1 presents the account postings specified in the trade-processing BRS; this indicates that the information generated through the trade-processing module is transferred to the fund-processing module.

Trade-Processing Account Postings

Upon initiation of the trade-processing process, accounting entries will be posted to the relevant accounts.

Processing of a ***buy*** trade for a non-custodial trade will result in following accounting entries.

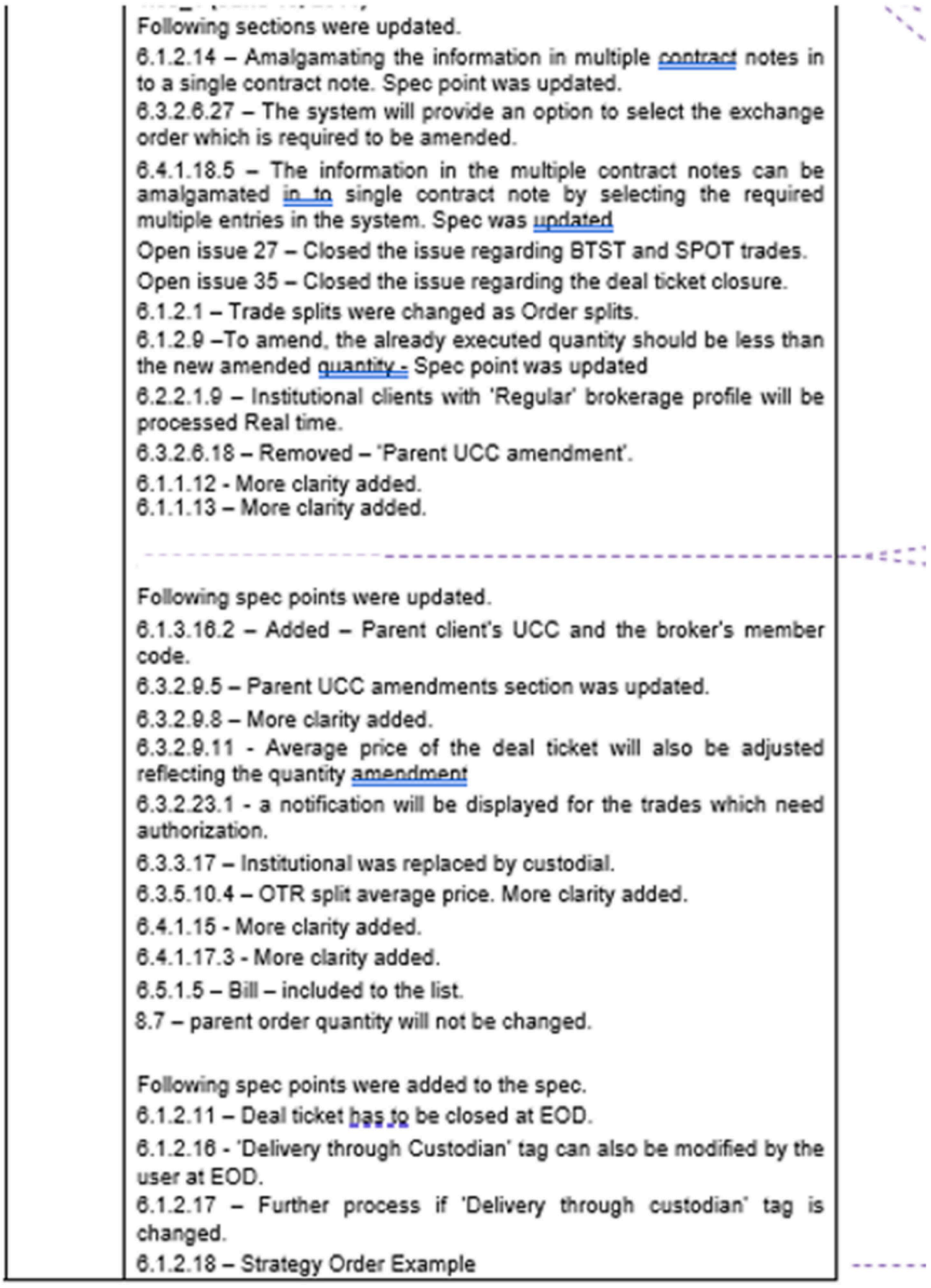
DR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX

Processing of a ***buy or sell*** custodial buy trade will result in the following accounting entries:

DR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX

Fig. J.1. Account postings in the trade-processing BRS.

Since the BRSs were frequently updated, it was difficult to establish outcome and behavior controls in project A. For example, the trade-processing BRS had eight versions. Although there were only 16 pages in the first version of the trade-processing BRS (i.e., V1.00), the eighth version of the trade-processing BRS (V1.08) consisted of 154 pages. Fig. J.2 illustrates a sample BRS revision history (page 8/154 of the trade-processing BRS (V1.08) _3i_final doc – project A).



Following sections were updated.

6.1.2.14 – Amalgamating the information in multiple contract notes in to a single contract note. Spec point was updated.

6.3.2.6.27 – The system will provide an option to select the exchange order which is required to be amended.

6.4.1.18.5 – The information in the multiple contract notes can be amalgamated in to single contract note by selecting the required multiple entries in the system. Spec was updated

Open issue 27 – Closed the issue regarding BTST and SPOT trades.

Open issue 35 – Closed the issue regarding the deal ticket closure.

6.1.2.1 – Trade splits were changed as Order splits.

6.1.2.9 –To amend, the already executed quantity should be less than the new amended quantity - Spec point was updated

6.2.2.1.9 – Institutional clients with 'Regular' brokerage profile will be processed Real time.

6.3.2.6.18 – Removed – 'Parent UCC amendment'.

6.1.1.12 - More clarity added.

6.1.1.13 – More clarity added.

Following spec points were updated.

6.1.3.16.2 – Added – Parent client's UCC and the broker's member code.

6.3.2.9.5 – Parent UCC amendments section was updated.

6.3.2.9.8 – More clarity added.

6.3.2.9.11 - Average price of the deal ticket will also be adjusted reflecting the quantity amendment

6.3.2.23.1 - a notification will be displayed for the trades which need authorization.

6.3.3.17 – Institutional was replaced by custodial.

6.3.5.10.4 – OTR split average price. More clarity added.

6.4.1.15 - More clarity added.

6.4.1.17.3 - More clarity added.

6.5.1.5 – Bill – included to the list.

8.7 – parent order quantity will not be changed.

Following spec points were added to the spec.

6.1.2.11 – Deal ticket has to be closed at EOD.

6.1.2.16 - 'Delivery through Custodian' tag can also be modified by the user at EOD.

6.1.2.17 – Further process if 'Delivery through custodian' tag is changed.

6.1.2.18 – Strategy Order Example

Fig. J.2. Sample BRS revision history (page 8/154 of the trade-processing BRS (V1.08)_3i_final doc – project A).

The evolution of specifications leads to unmanageable targets for the team members, leading to a minimum level of team spirit. According to respondent 03 from project A, "Ultimately, everybody came up with reasons to cover themselves. When I compare it to the other projects I worked on, teamwork was low." Due to these reasons, the project was dragged for three and one-half years. For example, respondent 02 mentioned that "It is

almost at the exit procedure level. So, we were [in] discussion how we get out from the project, because it has dragged for so long, three and half years." The analysis of BRSs confirmed that the project was at an unmanageable level. For example, the BRSs had several versions, some were shared with the client, some were internal versions, and even the business analysts were confused about the functions that had to be developed within relevant deadlines. During the observations, the first author observed tensions between team members, where they argued and passed the responsibilities between team members. The team members were not friendly with each other; they were concerned about the project performance and completing the assigned tasks on time. This indicated that the project was at an unmanageable level.

The above explanations indicate bidirectional dynamic information interdependency, where the dominant module (i.e., the trade-processing module) and the submissive module (i.e., the fund-processing module) rely partially or fully on each other for deriving information (i.e., "bidirectional") and the information interdependencies are not clearly identified (i.e., "undefined"). As explained above, it was difficult to establish formal controls and the evolution of specifications created unmanageable targets for the team members, leading to a minimum level of team spirit. Thus, outcome, behavior, and clan controls were identified as "auxiliary controls." As the functions were not well elaborated, team members had more self-control; thus, self-control was identified as "dominant control." Based on the interview data, observations, and examining the nature of BRS updates, the performance of project A was identified as "unstable."

We followed a similar approach in deriving three other types of information interdependencies (unidirectional permanent information dependency, unidirectional temporary information dependency, and bidirectional permanent information dependency), control portfolios, and performance of other projects.

References

[1] J. Gershenson, G. Prasad, Y. Zhang, Product modularity: definitions and benefits, J. Eng. Des. 14 (3) (2003) 295–313.
[2] A. Tiwana, Does technological modularity substitute for control? A study of alliance performance in software outsourcing, Strateg. Manage. J. 29 (7) (2008) 769–780.
[3] M. Cataldo, Dependencies in Geographically Distributed Software Development: Overcoming the Limits of Modularity, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2007.
[4] A. Lindberg, N. Berente, J. Gaskin, K. Lyytinen, Coordinating interdependencies in online communities: a study of an open source software project, Inf. Syst. Res. 27 (4) (2016) 751–772.
[5] L.G. Pee, A. Kankanhalli, H.-W. Kim, Knowledge sharing in information systems development: a social interdependence perspective, J. Assoc. Inf. Syst. 11 (10) (2010) 550–575.
[6] D. Stea, K. Foss, N.J. Foss, A neglected role for organizational design: supporting the credibility of delegation in organizations, J. Organ. Des. 4 (3) (2015) 3–17.
[7] B. Cho, S.Y. Ryoo, K.K. Kim, Interorganizational dependence, information transparency in interorganizational information systems and supply chain performance, Eur. J. Inf. Syst. 26 (2) (2017) 185–205.
[8] D. Dönmez, G. Grote, S. Brusoni, Routine interdependencies as a source of stability and flexibility. A study of agile software development teams, Inf. Organ. 26 (3) (2016) 63–83.
[9] V.D.V. Gerben, B. Emans, V.D.V. Evert, Effects of interdependencies in project teams, J. Soc. Psychol. 139 (2) (1999) 202–214.
[10] R.I. Swaab, M. Schaerer, E.M. Anicich, R. Ronay, A.D. Galinsky, The too-much-talent effect: team interdependence determines when more talent is too much or not enough, Psychol. Sci. 25 (8) (2014) 1581–1591.
[11] H. Mahmood, M.S. Rehman, Tools for software engineers, Int. J. Res. Eng. Technol. 3 (10) (2015) 75–86.
[12] A. Aurum, C. Wohlin, Engineering and Managing Software Requirements, Springer Science & Business Media, 2005.
[13] D. Mani, K. Srikanth, A. Bharadwaj, Efficacy of R&D work in offshore captive centers: an empirical study of task characteristics, coordination mechanisms, and performance, Inf. Syst. Res. 25 (4) (2014) 846–864.
[14] L.J. Kirsch, Portfolios of control modes and IS project management, Inf. Syst. Res. 8 (3) (1997) 215–239.
[15] W.G. Ouchi, M.A. Maguire, Organizational control: two functions, Adm. Sci. Q. 20 (4) (1975) 559–569.
[16] U. Remus, M. Wiener, C. Saunders, M. Mähring, The impact of control styles and control modes on individual-level outcomes: a first test of the integrated IS project control theory, Eur. J. Inf. Syst. (2020) 1–19.
[17] S.C. Srivastava, T.S.H. Teo, Contract performance in offshore systems development: role of control mechanisms, J. Manage. Inf. Syst. 29 (1) (2012) 115–158.
[18] A. Tiwana, Systems development ambidexterity: explaining the complementary and substitutive roles of formal and informal controls, J. Manage. Inf. Syst. 27 (2) (2010) 87–126.
[19] V. Choudhury, R. Sabherwal, Portfolios of control in outsourced software development projects, Inf. Syst. Res. 14 (3) (2003) 291–314.
[20] S. Karim, Modularity in organizational structure: the reconfiguration of internally developed and acquired business units, Strateg. Manage. J. 27 (9) (2006) 799–823.
[21] A. Cram, K. Brohman, B. Gallupe, Hitting a moving target: a process model of information systems control change, Inf. Syst. J. 26 (3) (2016) 195–226.
[22] A. Cram, K. Brohman, B. Gallupe, Information systems control: a review and framework for emerging Information Systems processes, J. Assoc. Inf. Syst. 17 (4) (2016) 216–266.
[23] M. Wiener, M. Mähring, U. Remus, C. Saunders, Control configuration and control enactment in INFORMATION SYSTEMS PRojects: review and expanded theoretical framework, MIS Q. 40 (3) (2016) 741–774.
[24] E.C. Cecil, M.D. Myers, Social control in information systems development: a negotiated order perspective, J. Inf. Technol. 33 (3) (2018) 173–187.
[25] L.M. Maruping, V. Venkatesh, R. Agarwal, A control theory perspective on agile methodology use and changing user requirements, Inf. Syst. Res. 20 (3) (2009) 377–399.
[26] K. Hölttä-Otto, O. de Weck, Degree of modularity in engineering systems and products with technical and business constraints, Concurr. Eng. 15 (2) (2007) 113–126.
[27] J.H. Mikkola, T. Skjøtt-Larsen, Supply-chain integration: implications for mass customization, modularization and postponement strategies, Prod. Plan. Control 15 (4) (2004) 352–361.
[28] P.J. Ågerfalk, B. Fitzgerald, H.H. Olsson, E.Ó. Conchúir, Benefits of global software development: the known and unknown. Making Globally Distributed Software Development a Success Story, Springer, 2008, pp. 1–9.
[29] J. Dedrick, E. Carmel, K.L. Kraemer, A dynamic model of offshore software development, J. Inf. Technol. 26 (1) (2011) 1–15.
[30] E.B. Allen, T.M. Khoshgoftaar, Measuring coupling and cohesion: an information-theory approach*IEEE*, in: Symposium Conducted at the Meeting of the Sixth International Software Metrics Symposium, 1999.
[31] M. Goulão, Coupling and cohesion as modularization drivers: are we being over-persuaded?, in: IEEE. Symposium Conducted at the Meeting of the Fifth European Conference on Software Maintenance and Reengineering, Lisbon, Portugal, 2001.
[32] K. Riemer, R.B. Johnston, Rethinking the place of the artefact in IS using Heidegger's analysis of equipment, Eur. J. Inf. Syst. 23 (3) (2014) 273–288.
[33] M. Aiken, J. Hage, Organizational interdependence and intra-organizational structure, Am. Sociol. Rev. (1968) 912–930.
[34] T. Elston, M. MacCarthaigh, K. Verhoest, Collaborative cost-cutting: productive efficiency as an interdependency between public organizations, Public Manage. Rev. 20 (12) (2018) 1815–1835.
[35] M. Ganesh, M. Gupta, Impact of virtualness and task interdependence on extra-role performance in software development teams, Team Perform. Manage.: Int. J. 16 (3/4) (2010) 169–186.
[36] Å.G. Dahlstedt, A. Persson, Requirements interdependencies: state of the art and future challenges. Engineering and Managing Software Requirements, Springer, 2005, pp. 95–116.
[37] K. Pohl, Process-centered Requirements Engineering, John Wiley & Sons, Inc, New York, USA, 1996.
[38] A. Caglio, A. Ditillo, Opening the black box of management accounting information exchanges in buyer–supplier relationships, Manage. Account. Res. 23 (2) (2012) 61–78.
[39] P. Carlshamre, K. Sandahl, M. Lindvall, B. Regnell, J.N.O. Dag, An industrial survey of requirements interdependencies in software product release planning, in: Presented at the Meeting of the Fifth IEEE International Symposium on requirements Engineering, Ontario, Canada, 2001.
[40] E.S. Gol, M.-K. Stein, M. Avital, Crowdwork platform governance toward organizational value creation, J. Strateg. Inf. Syst. 28 (2) (2019) 175–195.
[41] R. Sabherwal, The evolution of coordination in outsourced software development projects: a comparison of client and vendor perspectives, Inf. Organ. 13 (3) (2003) 153–202.
[42] A. Cram, M. Wiener, Technology-mediated control: case examples and research directions for the future of organizational control, Commun. Assoc. Inf. Syst. 46 (1) (2020) 4.
[43] M. Kreutzer, J. Walter, L.B. Cardinal, Organizational control as antidote to politics in the pursuit of strategic initiatives, Strateg. Manage. J. 36 (9) (2015) 1317–1337.
[44] S.V. Grabski, S.A. Leech, Complementary controls and ERP implementation success, Int. J. Account. Inf. Syst. 8 (1) (2007) 17–39.
[45] M. Keil, A. Rai, S. Liu, How user risk and requirements risk moderate the effects of formal and informal control on the process performance of IT projects, Eur. J. Inf. Syst. 22 (6) (2013) 650–672.
[46] A. Cram, M.K. Brohman, Controlling information systems development: a new typology for an evolving field, Inf. Syst. J. 23 (2) (2013) 137–154.

[47] R.W. Gregory, R. Beck, M. Keil, Control balancing in information systems development offshoring projects, MIS Q. 37 (4) (2013) 1211–1232.
[48] L.J. Kirsch, The management of complex tasks in organizations: controlling the systems development process, Organ. Sci. 7 (1) (1996) 1–21.
[49] M. Mähring, M. Wiener, U. Remus, Getting the control across: control transmission in Information Systems offshoring projects, Inf. Syst. J. 28 (4) (2017) 708–728.
[50] K.M. Eisenhardt, Control: organizational and economic approaches, Manage. Sci. 31 (2) (1985) 134–149.
[51] A. Tiwana, M. Keil, Does peripheral knowledge complement control? An empirical test in technology outsourcing alliances, Strateg. Manage. J. 28 (6) (2007) 623–634.
[52] M.A. Abubakre, M. Ravishankar, C.R. Coombs, The role of formal controls in facilitating information system diffusion, Inf. Manage. 52 (5) (2015) 599–609.
[53] W.G. Ouchi, The transmission of control through organizational hierarchy, Acad. Manage. J. 21 (2) (1978) 173–192.
[54] M. Wiener, U. Remus, J. Heumann, M. Mähring, The effective promotion of informal control in information systems offshoring projects, Eur. J. Inf. Syst. 24 (6) (2015) 569–587.
[55] C.C. Manz, H. Angle, Can group self-management mean a loss of personal control: triangulating a paradox, Group Organ. Stud. 11 (4) (1986) 309–339.
[56] J. Gerdin, The impact of departmental interdependencies and management accounting system use on subunit performance, Eur. Account. Rev. 14 (2) (2005) 297–327.
[57] U. Remus, M. Wiener, The amount of control in offshore software development projects, J. Glob. Inf. Manage. 20 (4) (2012) 1–26.
[58] M. Cataldo, J.D. Herbsleb, Coordination breakdowns and their impact on development productivity and software failures, IEEE Trans. Softw. Eng. 39 (3) (2013) 343–360.
[59] M. Cataldo, J.D. Herbsleb, K.M. Carley, Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity, in: Presented at the Meeting of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Kaiserslautern, Germany, 2008.
[60] J. Kotlarsky, Management of Globally Distributed Component-Based Software Development Projects, Erasmus University Rotterdam, 2005.
[61] R. Sanchez, J.T. Mahoney, Modularity, flexibility, and knowledge management in product and organization design, Strateg. Manage. J. 17 (Winter Special Issue) (1996) 63–76.
[62] C.E.H. Chua, W.-K. Lim, C. Soh, S.K. Sia, Enacting clan control in complex IT projects: a social capital perspective, MIS Q. 36 (2) (2012) 577–600.
[63] A. Tiwana, Does interfirm modularity complement ignorance? A field study of software outsourcing alliances, Strateg. Manage. J. 29 (11) (2008) 1241–1252.
[64] T. Holmqvist, M. Persson, Modularization-not only a product issue, in: Presented at the Meeting of the 7th Workshop on Product Structuring–Product Platform Development, Gothenburg, Sweden, 2004.
[65] W.G. Ouchi, A conceptual framework for the design of organizational control mechanisms, Manage. Sci. 25 (9) (1979) 833–848.
[66] R.W. Gregory, M. Keil, Blending bureaucratic and collaborative management styles to achieve control ambidexterity in IS projects, Eur. J. Inf. Syst. 23 (3) (2014) 343–356.
[67] M.E. Sosa, S.D. Eppinger, C.M. Rowles, The misalignment of product architecture and organizational structure in complex product development, Manage. Sci. 50 (12) (2004) 1674–1689.
[68] A. Tiwana, Evolutionary competition in platform ecosystems, Inf. Syst. Res. 26 (2) (2015) 266–281.
[69] H.K. Klein, M.D. Myers, A set of principles for conducting and evaluating interpretive field studies in information systems, MIS Q. 23 (1) (1999) 67–93.
[70] M.Q. Patton, Qualitative Research and Evaluation Methods, Sage, Thousand Oaks, Calif, 2002.
[71] R.M. Grinnell, P.A. Gabor, Y.A. Unrau, Program Evaluation For Social Workers: Foundations of Evidence-Based Practice, Oxford University Press, USA, 2015.
[72] N. Kumar, L.W. Stern, J.C. Anderson, Conducting interorganizational research using key informants, Acad. Manage. J. 36 (6) (1993) 1633–1651.
[73] V. Minichiello, R. Aroni, E. Timewell, L. Alexander, In-depth Interviewing: Principles, Techniques, Analysis, 2nd ed., Pearson Education Australia, Melbourne, Australia, 1995.
[74] G.A. Bowen, Document analysis as a qualitative research method, Qual. Res. J. 9 (2) (2009) 27–40.
[75] R.K. Yin, Case Study Research: Design and Methods, 3rd ed., Sage Publications, Thousand Oaks, CA, 2003.
[76] S. Chakraborty, S. Sarker, S. Rai, S. Sarker, R. Nadadhur, Offshore vendors' software development team configurations: an exploratory study, J. Glob. Inf. Manag. 19 (3) (2011) 1–29.
[77] D.R. Thomas, A general inductive approach for analyzing qualitative evaluation data, Am. J. Eval. 27 (2) (2006) 237–246.
[78] L.J. Kirsch, V. Sambamurthy, K. Dong-Gil, R.L. Purvis, Controlling information systems development projects: the view from the client, Manage. Sci. 48 (4) (2002) 484–498.
[79] S. Sarker, S. Sahay, Understanding virtual team development: an interpretive study, J. Assoc. Inf. Syst. 4 (1) (2003) 1.
[80] Y. Tim, S.L. Pan, S. Bahri, A. Fauzi, Digitally enabled affordances for community-driven environmental movement in rural Malaysia, Inf. Syst. J. 28 (1) (2018) 48–75.
[81] B. Sengupta, S. Chandra, V. Sinha, in: A Research Agenda for Distributed Software Development Symposium Conducted at the Meeting of the 28th International Conference on Software Engineering, Shanghai, China, 2006.
[82] R. Saavedra, P.C. Earley, L. Van Dyne, Complex interdependence in task-performing groups, J. Appl. Psychol. 78 (1) (1993) 61.
[83] M.S. Feldman, B.T. Pentland, Reconceptualizing organizational routines as a source of flexibility and change, Adm. Sci. Q. 48 (1) (2003) 94–118.
[84] Y. Wang, Y. Liu, C. Canel, Process coordination, project attributes and project performance in offshore-outsourced service projects, Int. J. Proj. Manage. 36 (7) (2018) 980–991.
[85] J.J. Jiang, G. Klein, J.C.-A. Tsai, Y. Li, Managing multiple-supplier project teams in new software development, Int. J. Proj. Manage. 36 (7) (2018) 925–939.
[86] L.J. Kirsch, Deploying common systems globally: the dynamics of control, Inf. Syst. Res. 15 (4) (2004) 374–395.
[87] D. Mougouei, Factoring requirement dependencies in software requirement selection using graphs and integer programming, in: Presented at the Meeting of the 31st IEEE/ACM International Conference on Automated Software Engineering, Singapore, 2016.
[88] A. Salado, R. Nilchiani, The concept of order of conflict in requirements engineering, IEEE Syst. J. 10 (1) (2016) 25–35.
[89] Sanchez, M.A Terlizzi, Cost and time project management success factors for information systems development projects, Int. J. Proj. Manage. 35 (8) (2017) 1608–1626.
[90] K. Siau, Y. Long, M. Ling, Toward a unified model of information systems development success, J. Database Manage. (JDM) 21 (1) (2010) 80–101.
[91] K.B. White, R. Leifer, Information systems development success: perspectives from project team participants, MIS Q. (1986) 215–223.
[92] T. Ahola, So alike yet so different: a typology of interorganisational projects, Int. J. Proj. Manage. 36 (8) (2018) 1007–1018.
[93] D. Baccarini, The concept of project complexity—a review, Int. J. Proj. Manage. 14 (4) (1996) 201–204.
[94] S. Ghobadi, What drives knowledge sharing in software development teams: a literature review and classification framework, Inf. Manage. 52 (1) (2015) 82–97.
[95] S. Ghobadi, J D'Ambra, Modeling high-quality knowledge sharing in cross-functional software development teams, Inf. Process. Manag. 49 (1) (2013) 138–157.
[96] I.B. Hong, A new framework for interorganizational systems based on the linkage of participants' roles, Inf. Manage. 39 (4) (2002) 261–270.
[97] D.W. Johnson, R.T. Johnson, New developments in social interdependence theory, Genet. Soc. Gen. Psychol. Monogr. 131 (4) (2005) 285–358.
[98] N. Parolia, J.J. Jiang, G. Klein, T.S. Sheu, The contribution of resource interdependence to IT program performance: a social interdependence perspective, Int. J. Proj. Manage. 29 (3) (2011) 313–324.
[99] F.T.C. Tan, B. Tan, W. Wang, D. Sedera, IT-enabled operational agility: an interdependencies perspective, Inf. Manage. 54 (3) (2017) 292–303.
[100] J.D. Thompson, Organizations in Action: Social Science Bases of Administrative Theory, 1st ed., Taylor & Francis Group, New York, 2017.
[101] S. Faraj, L. Sproull, Coordinating expertise in software development teams, Manage. Sci. 46 (12) (2000) 1554–1568.
[102] D.E. Strode, A dependency taxonomy for agile software development projects, Inf. Syst. Front. 18 (1) (2016) 23–46.
[103] P. Carlshamre, B. Regnell, Requirements lifecycle management and release planning in market-driven requirements engineering processes *IEEE*, in: Symposium Conducted at the Meeting of the 11th International Workshop on Database and Expert Systems Applications, Greenwich, London, U.K, 2000.
[104] B. Ramesh, M. Jarke, Toward reference models for requirements traceability, IEEE Trans. Softw. Eng. 27 (1) (2001) 58–93.
[105] Å.G. Dahlstedt, Requirements Interdependencies–a research framework. Software Change Impact Analysis, IEEE Computer Society Press, 2001.
[106] J. Li, L. Zhu, R. Jeffery, Y. Liu, H. Zhang, Q. Wang, M. Li, An initial evaluation of requirements dependency types in change propagation analysis, in: Symposium Conducted at the Meeting of the 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012), Ciudad Real, Spain, 2012.
[107] P.J. Gomes, N.R. Joglekar, Linking modularity with problem solving and coordination efforts, Manage. Decis. Econ. 29 (5) (2008) 443–457.
[108] C.A. Conley, L. Sproull, Easier said than done: an empirical investigation of software design and quality in open source software development, in: Presented at the Meeting of the Hawaii International Conference on System Sciences, Big Island, HI, USA, 2009.
[109] A. Narduzzo, A. Rossi, Modular design and the development of complex artifacts: lessons from free/open source software *Quaderno*, in: DISA, 80, Department of Computer and Management Sciences, University of Trento, Trento, Italy, 2003.
[110] A. Bhattacherjee, Social Science Research: Principles, Methods, and Practices, University of South Florida, 2012.
[111] M.E. Whitman, A.B. Woszczynski, The Handbook of Information Systems Research, Idea Group Publishing, 2004.

Maduka Subasinghage is a senior lecturer at Auckland University of Technology, New Zealand. She completed her PhD studies at the Queensland University of Technology, Australia. Prior to joining as a PhD candidate, she worked as a business analyst at a large software development organization. Her research interests include software development, modularization, outsourcing and knowledge management. Her research has been published in the *Information & Management Journal, Journal of Knowledge Management, Communications of the AIS, Enterprise Information Systems,* and *VINE: The Journal of Information and Knowledge Management Systems* and at international conferences such as the

International Conference on Information Systems (ICIS), the Pacific Asia Conference on Information Systems (PACIS), and the European Conference on Information Systems (ECIS).

Darshana Sedera has published his research in the information systems field in over 150 publications in major refereed journals and conferences. He researches the role of technology in innovation, entrepreneurship, and business transformation. He has strong methodological expertise in quantitative research methods. His publications have appeared in the *Journal of the AIS,* the *Journal of Strategic Information Systems, Information and Management,* and *Communications of the AIS.*

Shirish C. Srivastava is a professor and holds the GS1 Chair on "Digital Content for Omni Channel" at HEC Paris. His research interests include offshore sourcing, e-government, emerging technologies, and technology-enabled innovation. His research has been published in international refereed journals such as *MIS Quarterly, Information Systems Research, Journal of Management Information Systems, Journal of the Association for Information Systems, European Journal of Information Systems, Information Systems Journal, Journal of Information Technology, MIS Quarterly Executive, Communications of the AIS*, and *Journal of Global Information Management,* among others. He currently works as a senior editor at the *Journal of Association for Information Systems* and an associate editor at the *European Journal of Information Systems*.