Communications of the Association for Information Systems

Volume 46 | Article 29

6-2020

A Teaching Case on Information Systems Development Outsourcing: Lessons from a Failure

Subasinghage Maduka Nuwangi
Auckland University of Technology, maduka.subasinghage@aut.ac.nz

Darshana Sedera
Swinburne University of Technology

Recommended Citation

Nuwangi, S., & Sedera, D. (2020). A Teaching Case on Information Systems Development Outsourcing: Lessons from a Failure. Communications of the Association for Information Systems, 46, pp-pp. https://doi.org/10.17705/1CAIS.04629

Communications of the Association for Information Systems

Teaching Case DOI: 10.17705/1CAIS.04629 ISSN: 1529-3181

A Teaching Case on Information Systems Development Outsourcing: Lessons from a Failure

Subasinghage Maduka Nuwangi
Faculty of Business Economics and Law
Auckland University of Technology
New Zealand
maduka.subasinghage@aut.ac.nz

Darshana Sedera
Swinburne Business School
Swinburne University of Technology
Australia

Abstract:

Students and academics rarely receive an opportunity to investigate and learn from failed projects even though many organizations restrict access to information about failed projects in order to minimize reputational damage (Chua & Lam, 2005). However, failure cases can provide unique insights that one often ignores or cannot explore in successful projects (Lyytinen & Robey, 1999). To facilitate this learning, we present a teaching case based on an outsourced information systems development project that commenced in 2010 and was terminated in 2013. We observe the project's failure from the viewpoint of the vendor to illustrate how misspecified requirements and insufficient understanding of the client organization's specific requirements can lead to project failures. We derived the case description and analyses by conducting seven interviews with project team members and by analyzing 14 business requirement specification documents.

Keywords: Teaching Case, Project Failures, Information Systems Development, Outsourcing.

This manuscript underwent peer review. It was received 03/18/2019 and was with the authors for 4 months for 2 revisions. Carina de Villiers served as Associate Editor.

"Failure is simply the opportunity to begin again, this time more intelligently." – Henry Ford

1 Introduction

This teaching case facilitates discussion and learning via using a real-world information systems development (ISD) outsourcing project failure, which commenced in 2010 and was terminated in 2013. The client organization, a stock brokerage firm operating in India, was using a legacy system to manage the clearing and settlement of stock trades. The client was not satisfied with its legacy system and contacted the ISD outsourcing vendor to request it to develop an efficient information system.

At the beginning of the project, the client organization and the vendor's senior executives and line-of-business managers discussed the project scope and objectives. Once they agreed on the initial parameters and the project scope, senior managers assembled and assigned a team that comprised project managers, consultants, business analysts, technology leads, software engineers, system support engineers and quality assurance (QA) managers to the project. The consultants, business analysts, and project managers interacted with the client representatives to further specify the requirements, establish project timelines, and develop the project's budget. Next, the consultants and the business analysts transferred the extracted client requirements to the technology leads. The technology leads then informed the consultants and business analysts about the proposed system's technical requirements and the technical limitations. The vendor's QA team interacted with the business analysts and software engineers to test each part of the proposed system. The project management team also interacted with the vendor team's members to ensure that the vendor executed the project according to the derived project plan.

The project comprised four lifecycle phases: 1) initial discussions, 2) requirement engineering and design, 3) coding, and 4) testing. Figure 1 illustrates these phases and team members' involvement in each phase. While 21 team members worked on this project at the vendor organization, the client allocated only three employees from as points of contact. Despite the substantial efforts from both the client and vendor organizations, the client abandoned the project, which wasted substantial resources and jeopardized both the vendor's and client's reputation. Table 1 summarizes the project.

Educators can use this case study in IS or IS project management classes as a resource to discuss and understand the ISD process's pitfalls and complexities, ISD projects' fundamental elements, and project management more broadly. We derived the case discussion and analyses from seven interviews with project team members: a senior business analyst, project manager, specialist software engineer, two QA engineers, and two associate QA engineers. We also reviewed 14 business requirement specifications (BRSs) derived in the project. This teaching case focuses on the key issues that emerged due to the vendor's unintentionally misspecifying the client requirements. It also highlights the issues arising when a vendor does not clearly understand the specifications of a client's domain. Moreover, this study highlights the importance of risk management in ISD outsourcing projects. Overall, this case provides a real-world example of an ISD outsourcing project's failure and, thus, helps students understand how challenges at the requirement engineering stage could lead to project failures.

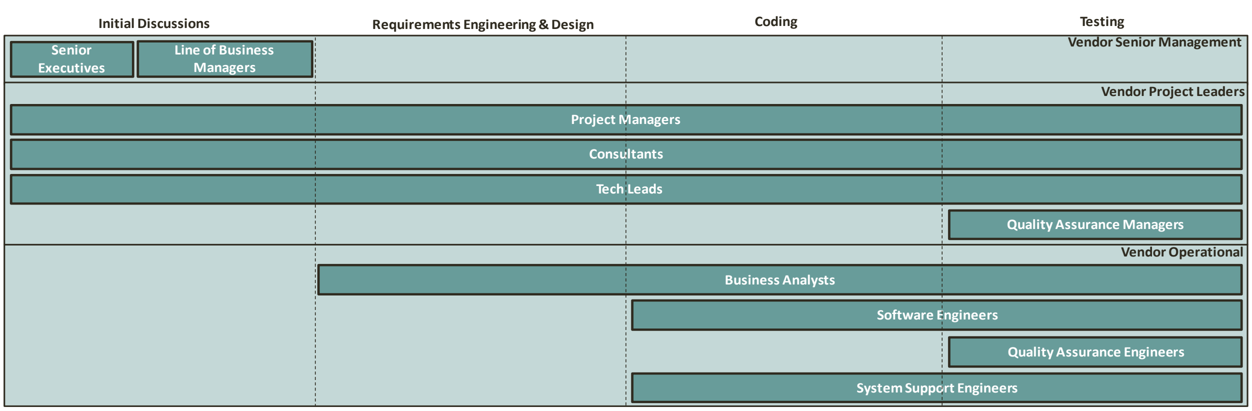


Figure 1. Vendor Organization Team Members' Involvement in the Project

Table 1. Summary of the Failed Project

Commissioned: 2010

Decommissioned: 2013

Client details: The client provided securities (e.g., equity, derivatives and online trading) brokerage services. The client organization had 112 branches in different cities across India; its main branch resided in Agra. The intention of the project was to develop software to manage the brokerage services that the client organization provided.

Vendor details: The vendor was a medium-sized software development company which specialized in the capital market domain. The vendor had been in the market for more than 10 years during which it delivered capital market solutions to more than 25 capital market clients all over the world.

Role and number of vendor-side employees on the project team:

- Senior management team (3): line-of-business manager, assistant vice-president of software development, and director of business operations
- Project management team (2): senior project manager and project manager
- Technical team (6): technology leads, associate technology leads, senior software engineers, and specialist software engineers
- Business analysis team (3): consultants and senior business analysts
- QA team (5): QA managers, QA engineers, and associate QA engineers
- System support engineer (1)
- User interface designer (1)

Reported key issues: 1) failure to clearly identify the client requirements, 2) failure to properly document the client requirements, 3) inaccurate estimations of system functionalities, 4) consultants made inaccurate decisions, 5) the technical team's lack of involvement during the requirement engineering process, and 6) lack of client commitment.

2 The Failed Project

2.1 The Client Organization

The client organization, a stockbroking firm operating in India, began operating in 2006 with the intention to become a leading financial intermediary that provided capital market access to investors. In the financial year ending in March, 2018, the client organization had ₹3,575,589,373 in total revenue and ₹2,571,210,788 in total expenses (in Indian rupees). The organization reported a ₹755,953,627 profit in the 2017-2018 financial year. The client organization provided services to various companies engaged in equity, derivative, and online trading (including financial institutions, corporations, and banks). The services that the client organization provided included: 1) mobile trading applications, 2) mobile alerts, 3) interactive charts, 4) offline and online trading applications, and 4) real-time information on trading asset classes all over the world. Furthermore, the client provided: 1) information on world stock indices (which covered the real-time quotes and movements of stock exchanges all over the world), 2) research reports that explained stock trading behaviors and news about derivatives and mutual funds, 3) updates on world news that may impact on stocks and trading behaviors, and 4) advice from stock market experts and an internal research team on trading stocks, currency, and commodities. The client organization dealt with multiple exchanges that had various asset classes, such as equities, securities lending, and borrowing. Each asset class comprised different market types such as: 1) normal markets (markets in which clients and vendors normally trade securities), and 2) auction markets (when clients cannot deliver the due shares for their sell trade, the exchange buys the shares of the relevant security in an auction market in order to deliver the shares and complete the buyer's obligation).

The client organization used a legacy system to manage stock trade clearing and settlements. However, it was not satisfied with the legacy system's performance in helping it to fulfill the complex services it provided. Thus, the organization initiated the ISD outsourcing project to obtain an efficient system that would enable it to manage the stock trade clearing and settlement and to provide better services to its customers.

2.2 The Vendor Organization

The vendor organization, a medium-sized ISD company, specialized in stock market-related ISDs and had more than 25 capital market clients all over the world. It began in 1996 and had more than 300 employees. The ISD outsourcing vendor provided IS solutions such as: 1) exchange solutions, 2) surveillance solutions, 3) smart order routing solutions, 4) post-trade solutions, and 5) market data solutions. The IS solutions provided a company with the ability to trade using multiple asset classes such as equities, securities lending, and borrowing[1]. The IS solutions included functionalities for multiple trading methods, such as auctions and continuous matching[2]. Furthermore, the systems provided a company with the ability to trade in multiple market structures such as regulated exchanges and over-the-counter markets.

Rather than developing IS solutions for each client from scratch, the vendor organization reused parts of its existing software products to develop new solutions. The vendor employed five types of previously developed software products in combination: 1) exchange products, 2) surveillance products, 3) smart order routing products, 4) post-trade products, and 5) market data products. The vendor also had some specialization in tailoring its existing generic solutions to specific client requirements in a short period. In addition to the aforementioned systems, the vendor also offered consultancy services and IT infrastructure services. To perform this role, the vendor had several partners all over the world who provided the required outsourcing consultancy services, hardware, and software services. Moreover, the vendor had specialized industry-specific teams that interacted with internal (e.g., marketing department) and external (e.g., hardware providers) stakeholders to develop ISD solutions and to better deliver client solutions.

2.3 The Project

The project specifically focused on developing a post-stock trade application for clearing and settling stock trades in a stock exchange. Naturally, the solution would reflect the complexities associated with stock trade-processing methods. Software application would have the following key functionalities:

- Trade processing: facilitate the entire trade lifecycle, which includes trade entries, trade amendments, trade splits, trade confirmations, trade rejections, and the printing of contract notes.
- User management: categorize users and assign roles to them; the user should receive the system privileges relevant to the assigned role.
- Fund processing: process transactions through ledger accounts and provide current cash positions to clients and brokering firms.
- General accounting and journal entries: capture transactions related to trade processing and general brokering activities.
- Stock processing: facilitate the processing of stocks when 1) clients deliver shares to the brokering firm, 2) the broker delivers shares to the exchange, 3) the exchange delivers shares to the broker, and 4) the broker delivers shares to clients.

When the project began, the vendor had a post-stock trade software product that it planned to adapt. However, this software had not reached maturity or become well established. Therefore, during the project, the team members had to develop most of the post-stock trade software's functionalities from scratch. The vendor employees lacked awareness of post-trade products' and solutions' specific functionalities. As such, they had to learn and develop most functionalities from scratch without much support.

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

[2] Auction trading involves calculating the opening and closing prices of a security at the opening and closing of trading hours, whereas continuous matching operates during the regular trading sessions.

2.3.1 The Project Team

Figure 2 and Table 2 provide details about the project's structure and the team members' tasks. The vendor assigned team members to four groups (i.e., group A: project management team, group B: business analyst team, group C: technical team, and group D: QA team) according to their tasks and responsibilities. These team members had experience and been involved in multiple successful projects in the vendor organization. For example, the consultants and senior business analysts engaged in a separate ISD project that they later completed successfully. Thus, the vendor organization deemed that the team members could plan, analyze, and deliver the project successfully.

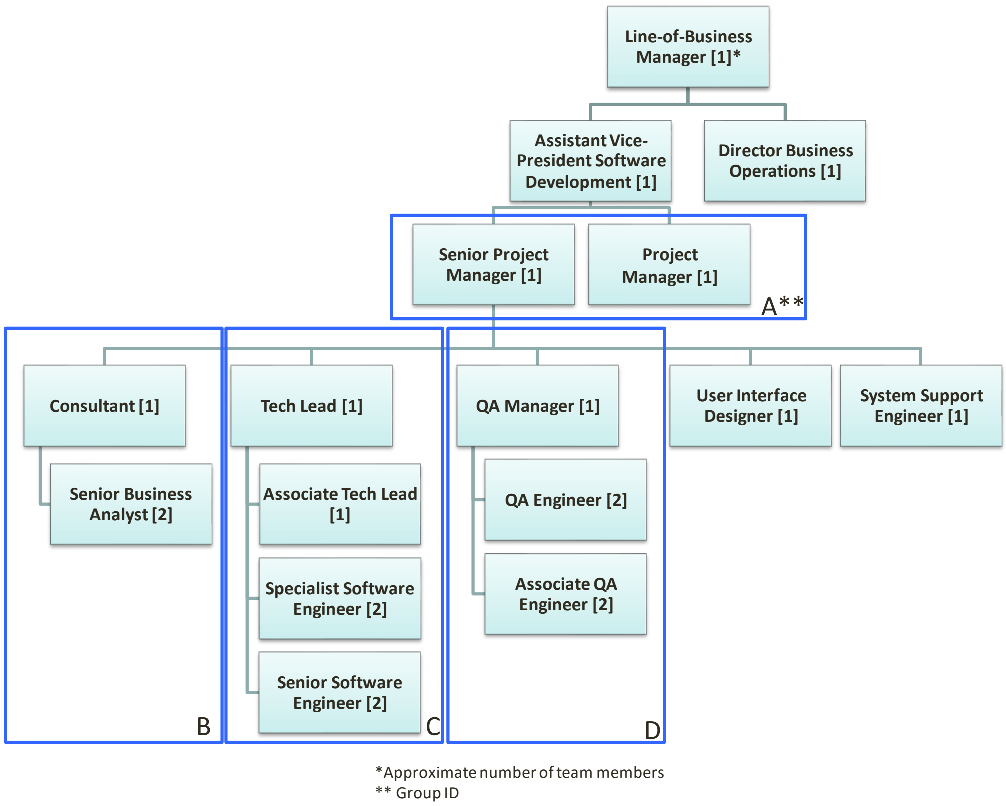


Figure 2. Project Structure (Numbers in Square Brackets Indicate the Number of Employees)

Table 2. Team Members' Responsibilities

Designation	Description[3]
Line-of-business manager	The vendor organization assigned the line-of-business managers to develop two products in the vendor organization: 1) smart order routing products and 2) post-trade products. Their responsibilities included designing, pricing, and managing product promotions. The line-of-business managers supervised the assistant vice-president of software development and director of business operations. The line-of-business manager on the project had approximately 10 years' project experience in the organization.
Assistant vice-president of software development	The assistant vice-president of software development managed all the software development activities in the vendor organization. The assistant vice-president also served as the head of the software development team in the vendor organization. The assistant vice-president of software development had approximately 18 years' experience in the organization.
Directors of business operations	The vendor organization assigned the directors of business operations to develop: 1) exchange products, 2) surveillance products, 3) smart order routing products, and 4) post-trade products. Their responsibilities included managing all the solutions in a product. Each director of business operations had over 10 years' experience in the organization.
Senior project manager	The vendor organization assigned a senior project manager to multiple solutions in a specific product. This individual's responsibilities included planning, monitoring, and controlling deliverables and work allocations. The senior project manager on the project had seven years' experience in the ISD industry.
Project manager	The project manager assisted the senior project manager in managing deliverables, allocating work, and project tracking. The project manager on the project had worked in the ISD industry for 15 years as a programmer, associate program manager, head of product delivery, and senior technical consultant. The project manager joined the project in 2012.
Consultants	Consultants assisted project managers in defining a solution in the project's agreed scope. Their responsibilities included client-requirement analysis and documentation. The consultants working on the project had an average of nine years' work experience.
Senior business analysts	The senior business analysts' responsibilities included analyzing and documenting client requirements. The senior business analysts sought advice from consultants when required. One of the senior business analysts on the project had six years' experience in the organization and several years' experience in the ISD industry sector.
Technology lead	The technology lead designed and planned the software development activities and guided the software engineering team in developing software based on the client requirements. The technology lead also reviewed the team members' software code to ensure its consistency and quality across the teams. The technology lead on the project had approximately 15 years' experience in the organization.
Associate technology lead	The associate technology lead supported technology leads in designing and planning software development activities. The associate technology lead on the project had approximately 10 years' experience in the ISD industry.
Specialist software engineers	Specialist software engineers conducted software development activities such as software coding and unit testing. On average, the specialist software engineers in the organization had approximately eight years' experience in the ISD industry.
Senior software engineers	Senior software engineers conducted software development activities such as software coding and unit testing. They received assistance from specialist software engineers and technology leads when required. On average, the senior software engineers in the organization had approximately six years' experience in the ISD industry.
QA managers	The vendor organization outsourced the project's QA functions to another ISD company, so it hired one QA manager to govern the outsourcing process. The QA manager governed the QA functions that QA engineers conducted. The QA manager's responsibilities included ensuring that the QA engineers completed the QA functions on time, identifying issues in the QA process, and transferring information to the project management team. The QA manager on the project had six years' experience at the vendor company. The QA manager had worked in the industry for 15 years in several companies as a senior QA engineer, a project manager, a test manager, and a program manager.

[3] The information presented in the table was verified through LinkedIn.

Table 2. Team Members' Responsibilities

QA engineers	The QA engineers conducted the QA process, which included documenting test scenarios and test cases, gathering test data, and conducting various quality assurance tests (e.g., integration testing, regression testing, and acceptance testing). One QA engineer was a fresh graduate who worked on the project for 1.5 years. The other QA engineer worked on the project for two years and had worked in the ISD industry for a year before joining the project.
Associate QA engineers	The associate QA engineers conducted the QA process. The QA engineers supported the associate QA engineers by providing assistance as required during the QA process. The two associate QA engineers on the project were fresh graduates and worked on the project for two years.
User interface designer	The user interface designers illustrated user interfaces through storyboards and process flows. They designed various user interfaces including menus, tabs, and widgets. The user interface designer on the project was a fresh graduate and worked on the project for two years.
System support engineer	The responsibilities of system support engineers included administering databases, system monitoring, and supporting the technical team in the ISD process. The system support engineer on the project was a fresh graduate and worked on the project for three years.

3 Early Signs of Problems

The requirement engineering process that the vendor organization conducted comprised several related phases: 1) requirement elicitation; 2) requirement analysis, modeling, and communication; 3) agreement on requirements; and 4) requirement evolution. During the requirement-elicitation stage, project team members (i.e., a consultant, project manager, and technology lead) visited the client premises to conduct initial discussions. During the initial discussions, the team representatives and the client agreed on the high-level project goals, timeline, and budget.

3.1 Decomposition and Documentation of the Requirements

After the client visit, the consultant team commenced requirement analysis, modelling, and documentation. During this stage, the consultants created process flow diagrams to explain the process flows of the main functionalities trade processing, fund processing, and stock processing. Based on the process flows, the consultants could then initially document the BRSs. The business analysts decomposed the client requirements into 11 requirement modules, which 11 separate BRSs described. Each BRS comprised information about the requirement modules, such as functionalities, dependencies, parameters, and concepts. For example, the trade processing BRS explained the following procedure: entering the trade into the system, processing the trade, managing the trade, generating the contract and bill, confirming the trade, and rejecting the trade. Each BRS in the project contained 50 to 100 pages. The consultants mainly presented the requirements in text with diagrams, account postings, and tables where necessary. They also provided example calculations and account postings for complex transactions. The business analysts also documented mind maps to illustrate the process flow of each requirement module.

3.2 Application of an Incremental and Iterative Development Method

The project followed the incremental and iterative development method (Kneuper, 2018). Kneuper (2018) defines the incremental and iterative lifecycle as:

> *A project life cycle where the project scope is generally determined early in the project life cycle, but time and cost estimates are routinely modified as the project team understanding of the product increases. Iterations develop the product through a series of repeated cycles, while increments successively add to the functionality of the product.* (p. 90)

Moreover, the project followed a "release train" approach whereby team members divided ISD tasks into multiple iterations in short time frames and delivered some proportion of the system to the client at the end of each iteration (Boehm, 1988). As specified in the "spec review schedule" document (see Table 3), the vendor and client planned for the vendor to deliver system's requirement modules in three deliverables (drops): 1) client registration, trade processing, brokerage, taxes and charges, users and user management, and master data; 2) fund processing, stock processing, general accounting and journal entries, depository participant module, and master data; and 3) derivatives, initial public offerings, manual funds processing, and master data. In each deliverable, the vendor sent BRSs to the client multiple times for review. For example, the client reviewed the client registration BRS twice before the sign-off. Since the client did not reside in India like the vendor did, the two companies had few face-to-face meetings.

Table 3. Example of a Specification Review Schedule Document with Initially Planned Dates

Spec ref	Specification	Drop	Review [client]	Review inc. [vendor]	2nd review [client]	Review inc. [vendor]	Sign-off [client]
Vol 01	Client Registration	1	Done	Done	17-Mar-11	23/24-Mar-11	28-Mar-11
Vol 03	Trade Processing	1	Done	Done	23-Mar-11	25-Mar-11	29-Mar-11
Vol 06	Brokerages, Taxes and Charges	1	Done	Done	21-Mar-11	28-Mar-11	30-Mar-11
Vol 10	Users and User Management	1	Done	Done	17-Mar-11	23-Mar-11	25-Mar-11
Vol 02	Master Data [Drop 1]	1	Done	Done	25-Mar-11	29-Mar-11	31-Mar-11
Vol 04	Fund Processing	2	30-Mar-11	1-Apr-11	5-Apr-11	6-Apr-11	8-Apr-11
Vol 05	Stock Processing	2	31-Mar-11	5-Apr-11	6-Apr-11	7-Apr-11	8-Apr-11
Vol 07	General Accounting and Journal Entries	2	1-Apr-11	4-Apr-11	7-Apr-11	8-Apr-11	11-Apr-11
Vol 11	Depository Participant Module	2	4-Apr-11	5-Apr-11	7-Apr-11	11-Apr-11	12-Apr-11
Vol 02	Master Data [Drop 2]	2	5-Apr-11	6-Apr-11	8-Apr-11	11-Apr-11	13-Apr-11
Vol 08	Derivatives (Futures and Options)	3	11-Apr-11	15-Apr-11	18-Apr-11	20-Apr-11	22-Apr-11
Vol 09	Initial Public Offering and Manual Funds Processing	3	12-Apr-11	18-Apr-11	19-Apr-11	20-Apr-11	22-Apr-11
Vol 02	Master Data [Drop 3]	3	15-Apr-11	19-Apr-11	20-Apr-11	22-Apr-11	25-Apr-11

3.3 Client Involvement in the Initial Stages of the Project

The client and vendor communicated through video conference calls and emails. The consultant team held a BRS finalization video conference call with the client before the client organization signed off all the BRSs. During this BRS finalization call, the vendor organization's consultant team clarified any doubts about the project's scope and functionality. The vendor organization shared the finalized BRSs with the technical team only after the client signed them off.

Consultants of the vendor organization and client representatives referred to any unclear concepts as "open issues" in the BRSs. Both the client and the vendor had the ability to note unclear concepts so that the other party could provide feedback on how to resolve them. Consultants or client representatives documented the communications between the client and vendor team regarding these open issues in the BRS (see Table 4). In some situations, the client did not provide clear answers (see italicized text in Table 4). As a result, consultants or client representatives closed the open issues only after several rounds of communication.

The teaching case focuses on the three BRSs that the authors identified in the project as the required system's main functionalities: trade processing, fund processing, and stock processing. The vendor completed the first iteration of the requirement engineering process in November, 2010, with the following outputs: the trade processing BRS (V1.00), the fund processing BRS (V1.00), and the stock processing BRS (V1.00). Consultants of the vendor organization shared the signed-off BRSs with the technical team for development.

Table 4. Examples of Open Issues

Date	Description	Owner	Status
15 October, 2010	Notification of restricted securities by the exchange: 1) How should restricted securities be identified in the security master file (i.e., which field is updated)? a) The securities master file provided by the exchange has the status flag of active/suspended. 2) What should the system do for the trade of security which is in the restricted list? (pls define restricted list) We presume that…will take preventive action regarding the same. (vendor to revert) **[Client]** …[4]site gives the list of…banned list when the trade is received for…or…clients and the banned scrip is being entered in…for execution, the list should be flashed on the screen. same should be the case for restricted securities. **[Vendor]** When restricted security is received by the [project] from […], the specific trade will be notified to the user. *3) Is the "Restricted Securities" file sent from the Exchanges AND the Depositories or only from the Exchanges?* *No Clear.* *(Are we talking about banned scrips)* **[Vendor]**: …will take preventive action to ensure orders are not submitted for inactive securities. Not a [project] functionality. Active/inactive status will be available in the security file provided by the exchanges, and the [project] will be updating the security master accordingly. *Please confirm.* **[Client]** Restricted and Inactive securities, both are different. We are talking about the […] banned list which is available on the…and…site. The site covers the permissible limit of scrip/6 series trading. **[Vendor]** Banned security check (exchange banned securities, …banned securities and…banned securities) needs to be carried out by the…prior to order submission. SOR [smart order routing] will confirm the feasibility once their current CR's are finalized. **[Client]** The restricted and inactive will have to be taken care of which is received from exchanges files. The list of scrips published by…and even displayed on…site should be monitored by…team before executing the orders and if at all executed should be highlighted in the…system. **[Vendor]** …system would indicate these banned trades if they received via the exchange file.	[Vendor]	Closed

The signed-off BRSs in the first round of requirement gathering supported the second round of the requirement engineering process. After the second round of requirement elicitation, in January, 2011, the vendor team updated the initial BRSs trade processing, fund processing, and stock processing as per the client's comments. The vendor team added summaries of the client feedback and BRS updates in the revision history of each BRS. In situations where the vendor team received extensive feedback from the client and updated the BRSs extensively, the revision history simply stated that the vendor team updated the entire BRS as per the client comments. For example, in the revision history of the trade-processing BRS, the vendor team noted: "Jan 21, 2011 - Trade Processing (V1.01) - Updated with [client][5] comments (Entire document updated).".

The vendor team sent the updated BRSs to the client for review; thus, the client had an opportunity to reevaluate whether the vendor team had integrated its feedback into the BRS. In the situations where the

[4] We conceal sensitive information in the table in order to maintain confidentiality.
[5] We add clarifications to extracts as necessary (in square brackets).

client was not satisfied with updates, the client included feedback as comments in the BRS and sent it back to the consultant team. The consultation team and client also conducted conference calls when necessary. This process continued until the client and the consultant teams agreed with the requirements specified in each BRS.

The vendor team completed the second iteration of the requirement engineering process in January, 2011, with the following outputs: the trade processing BRS (V1.01), the fund processing BRS (V1.01), the stock processing BRS (V1.01), the trade processing flow diagram (V1.00), the fund processing flow diagram, and the stock processing flow diagram (V1.01).

3.4 Consultant Visit to Client Site

During the third iteration, a lead project consultant remained at the client site for one month. During this period, the consultant continued requirement elicitation, analysis, and BRS documentation. The following record appeared in the revision history of the trade processing BRS: “March 10, 2011 – Trade Processing (V1.02) Updated spec [specification] with feedback received from the visit to [the client].”.

However, business analysts deemed one month to gather requirements at the client site insufficient due to the project’s substantial scope. A senior business analyst explained:

> *The problem with the requirement gathering was in the initial part. The time given to her was a month. ...One month is not enough. ...Scope is very large. ...There were business requirement specifications after that month, but as the project progressed, we realized...the requirements were not very clear.*

Business analysts established that they had not clearly identified the client requirements during the requirement-elicitation stage. The client and the consultant teams had discussions during initial meetings and established the broad scope of the required system’s functionalities. However, the technical team did not review the draft BRSs before sign-off. A senior business analyst explained: “We should have given the BRSs before signing off for the development to review. It never happened.”.

As a result, the technical team could not provide information about the system’s technical limitations before signing off the BRSs.

4 Escalation of Problems

During the project’s initial stages, the client and the vendor could have evaluated the likelihood that it would obtain ambiguous requirements and an uncertain scope. Therein, they could have allocated the appropriate time and resources necessary to gather requirements in detail. However, the vendor team realized the system’s complexity only after the technical team began the coding process. When the technical team began this process, it had to identify all the processes in greater detail. Identifying these processes required much more time and frequent interactions with the client than the team anticipated in the project’s initial stages.

4.1 The Level of Information Specified in the BRSs

Although the BRSs comprised high-level functional requirements, they lacked information about the system’s implementation procedures. A senior business analyst explained: “They signed off the business functionality in a basic way. BRSs don’t have this exactly how we are going to give [this] to you.”.

After the client agreed on and signed off the BRSs, the consultants updated the BRSs including the implementation details. However, the consultants did not inform the client about these updates. According to a senior business analyst:

> *We did the changes on the top of signed BRSs. ...We couldn’t update the client. So, the signed-off BRSs are like one set; we have a new set of BRSs which are something different from the signed-off BRSs.*

Because the BRSs did not include detailed information about the requirements, the QA team encountered difficulties during testing. A QA engineer explained: “Sometimes we don’t know whether it is a defect or not. We don’t have proper BRSs. That is the major problem.”.

Because the signed-off BRSs did not include clear information, the client requested the vendor add several new adjustments in the requirements even after the vendor and client has signed off the BRSs. A project manager explained:

> *The client [is] so demanding, whatever requirement they find out today, they put into the go live scope. They say "ok, this should be there, it is an obvious thing it should be there". Initially, we [the ISD team] didn't identify, didn't realize [the difficulty of providing the functionalities].*

4.2 Client Involvement in the Project

Although the client demanded new features in the project's latter stages, the client did not respond to emails requesting requirement clarifications that the vendor team sent. A senior business analyst said: "Their [the client's] commitment was minimum. So, when we even ask a simple question they didn't reply.".

The client did not respond because it had no dedicated team for the project. Team members assigned to the project had to manage the project while they performed their day-to-day operations. A senior business analyst mentioned:

> *There was no particular person dedicated to the project. They were doing the project concurrently with their day-to-day operations. Because of that, they didn't have time to answer our queries. So, if I sent an email, asking for clarifications, it took around one week to get it back.*

The senior business analyst continued: "We were trying to contact one person, and he was not available most of the time.".

As a result, the consultant team had to find alternative methods to elaborate on the client requirements. A senior business analyst reflected: "It was very difficult, sometimes we came up with solutions and couldn't wait for the client's reply. We have to come up with the solutions, and we have to elaborate on the functionalities.".

Due to client's poor commitment, consultant team members defined system's features based on their knowledge and understanding of the client requirements. However, the client never used some software functionalities that the consultant team suggested. A senior business analyst added: "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. So, that was a big problem.".

In order to potentially sell the system to other clients, the consultant team included new features in the system even without the client's request. A senior business analyst explained: "Most of the time what we did was, we added some alternatives. For example, if the client wants one and two [features], we added three and four.".

4.3 Communication between the Consultants and the Technical Team

The vendor organization sent one member from the technical team for the initial client visit. Other than that, the consultants and the technical team in the vendor organization minimally communicated about the requirements engineering process. As a result, the technical team could not suggest or discuss the necessary system functionalities with other team members. When the consultants suggested a solution for a specific issue in the requirements, the technical team rarely agreed with the consultants' suggestions. A senior business analyst stated: "Most of the time, when we [the consultants] suggest a solution, [the technical team said] we can't do this.".

This unproductive communication highlights the lack of understanding between the consultant team and the technical team.

4.4 Impact of the Interdependencies between Requirements

Because the requirements contained substantial overlaps and interdependencies, a change in one BRS cascaded to changes in other BRSs as well. For example, the system needed to calculate funds based on the executed trades. Thus, the "fund processing" feature pertained to the trade processing module as well. Rather than including all the fund processing requirements only in the fund processing BRS, consultants also included some of the fund processing requirements in the trade processing BRS, which meant that a change in the requirements in the fund processing BRS resulted in changes in the trade processing BRS as well. Since the technical team could not implement some initially agreed requirements due to the interdependencies between the requirement modules, consultants had to remove those requirements later in the ISD lifecycle. A senior business analyst stated: "This requirement cannot be implemented without that [requirement], because it has clashed with another requirement. So, a big requirement was removed.".

As the revision history of the fund processing BRS specified, on 5 December, 2011, consultants had to remove the system's accounting structure because the technical team could not implement it due to the interdependencies between the requirement modules. Because consultants removed the accounting structure, they 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 towards the end of the lifecycle. For example, on 13 August, 2012, consultants updated the fund processing BRS by: 1) adding or deleting 85 specification points, 2) amending 33 specification points, and 3) clarifying a further 31 specification points. Table 5 demonstrates the number of updates in the fund processing BRS. Appendix A provides a sample of the revision history.

Table 5. Number of Updates in the Fund Processing BRS

Fund processing			
Date	**Number of spec point updates**		
	Added/deleted*	Amended**	Clarified***
7 June, 2011	6	8	0
14 June, 2011	1	6	2
22 June, 2011	19	20	8
5 December, 2011	116	73	29
26 February, 2012	7	3	1
3 March, 2012	21	15	0
31 May, 2012	26	14	0
13 August, 2012	85	33	31
27 February, 2013	20	6	0

Note: * A spec point was added or deleted from BRS.
**: A spec point was amended.
***: A spec point was clarified including clarification information.

Because the trade processing BRS had interdependencies with the fund processing BRS, consultants needed to update the trade processing BRS as well. Due to the updates to the fund processing BRS, consultants added 92 trade processing specification points, amended 35 points, and clarified a further 26 points on 2 February, 2012. Table 6 demonstrates the number of updates in the trade-processing BRS.

Table 6. Number of Updates in the Trade Processing BRS

Trade processing			
Date	**Number of spec point updates**		
	Added/deleted	**Amended**	**Clarified**
20 September, 2011	38	10	0
31 October, 2011	19	5	3
2 February, 2012	92	35	26
2 March, 2012	2	14	0
25 June, 2012	13	15	0
25 July, 2012	37	37	0

From the information that we present above, we can infer that an error in how the vendor identified interdependencies or clashes between requirement modules caused fluctuations in BRSs towards the later stages of the project lifecycle. As a result, software engineers had to update the software code. A project manager stated:

> *They [the software engineers] have to change certain things because the document [BRS] is changing; it is changing continuously. It is frequently changing. So, the developer [software engineers] can't always accommodate the changes…. It is not easy. It is not a proper practice.*

In further exploring the BRSs, we found that some BRSs shared requirements with other BRSs. For example, the fund processing module had interrelationships with the trade processing module in that the trade processing BRS also included requirements related to the fund processing module. For example, the trade processing BRS included the "non-custodial buy trade" function and the "custodial buy trade", which the fund processing BRS also included. Figure 3 presents the account postings specified in the trade processing BRS[6].

Similarly, although the trade processing BRS mentioned some trade processing account postings (custodial and non-custodial), the fund processing BRS mentioned other account postings related to trade processing. Figure 4 presents the account postings that the fund processing BRS specified.

This process parallels the component-sharing modularity that Pine (1993) suggests whereby multiple modules share the same component. Similarly, in the ISD project we investigated, multiple BRSs shared the same requirements (see Figure 5). Therefore, a change in requirements in one BRS created several changes to other BRSs.

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.

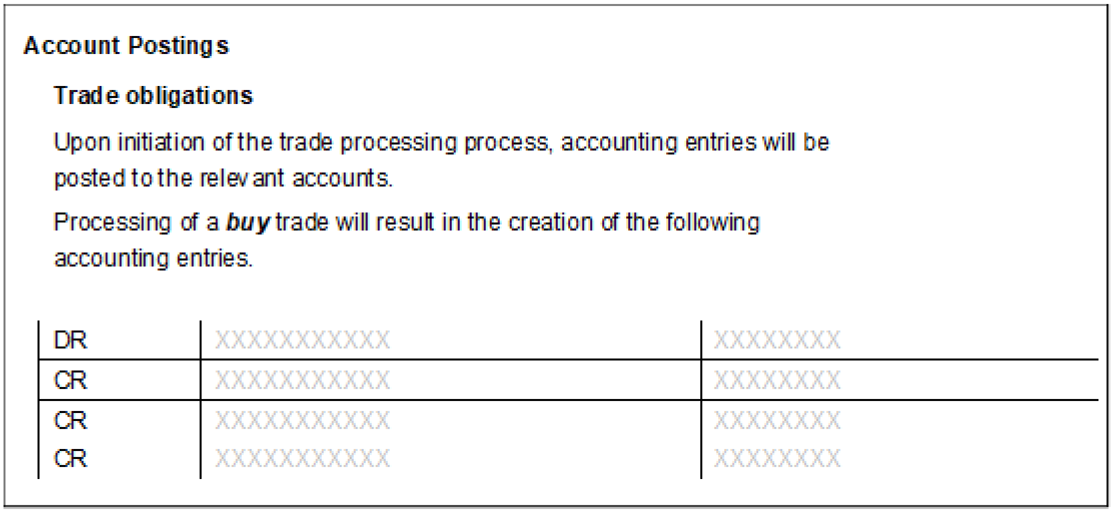
DR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX
CR	XXXXXXXXXXX	XXXXXXXX

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

Figure 3. Account Postings in the Trade Processing BRS

[6] We conceal sensitive information in the figure in order to maintain confidentiality.



Account Postings

Trade obligations

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

Processing of a ***buy*** trade will result in the creation of the following accounting entries.

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

Figure 4. Account Postings in the Fund Processing BRS

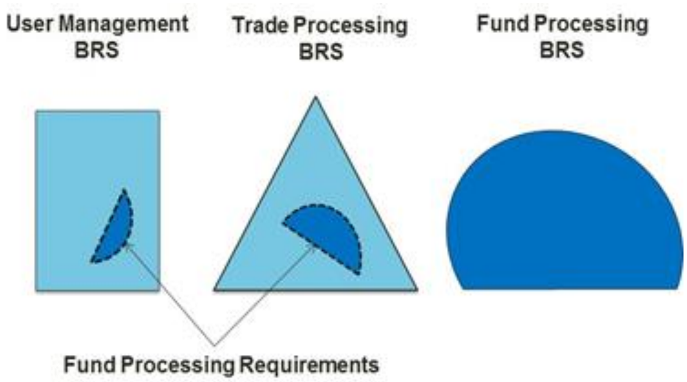


Figure 5. Component-sharing Modularity

4.5 Final Stages of the Project

Following the "release train" method (Kneuper, 2018), the vendor team tried to deliver components according to the previously agreed timeframe. However, as the project progressed, the team struggled to complete the deliverables on time. On some occasions, the team had to deliver whatever components it had been completed at the time. Because the project went through multiple iterations, the team updated BRSs frequently. For example, the trade processing BRS had eight versions. Although the first version of the trade processing BRS (i.e., V1.00) comprised only 16 pages, the eighth version (i.e., v1.08) comprised 154 pages. A senior business analyst explained: "For like two years the BRSs have been frequently updated. So, after like two years we had 95% complete set of BRSs. Initially, that was not completed.".

The requirement engineering iterations took place over three years. A project manager reported: "It is almost at the exit procedure level. So, we were [in] discussion [about] how we get out from the project.".

A senior business analyst commented: "We have temporarily halted it, but we haven't stopped it. We are looking for another potential client. If we get one, we will open it again.".

5 Summary of the Key Issues

Table 7 summarizes the key issues at the four stages of the requirement engineering process.

Table 7. Summary of Key Issues in the Requirement Engineering Process

Stage	Description	Key issues of the requirement engineering process
Requirement elicitation	Consultants failed to anticipate risks associated with requirement ambiguity and uncertainty of the scope	Senior management team did not allocate sufficient time to gather requirements
	Client requirements lacked clarity	Consultants did not clearly identify the client requirements
	The client never used some requirements that the consultants suggested	Consultants inaccurately estimated the system's functionalities
	Consultant team included new requirements even without client request	Consultants inaccurately estimated the system's functionalities Consultants did not sufficiently understand client requirements Consultants made inaccurate decisions
Requirement analysis modeling and communication	Insufficient information in BRSs	Consultants did not properly document the client requirements Lack of client commitment
	Consultants updated documented BRSs multiple times including clarification information	Consultants did not properly document the client requirements
	Consultants updated signed-off BRSs	Consultants made inaccurate decisions
	Technical team disagreed with the consultants' suggestions	The consultant team and technical team did not adequately understand client requirements
Agreement on the requirements	The technical staff did not review BRSs before sign-off	The technical team did not sufficiently participate in requirement gathering
	Many disagreements during BRS finalization calls	Team spirit between the client and consultant team declined gradually
Evolution of requirements	Client requested several new requirements after it had signed-off on BRSs	Client continued to make demands
	Consultants removed some requirements from the BRSs	Consultants inaccurately estimated system functionalities Consultants lacked understanding about the interdependencies and clashes between modules

6 Conclusion

Educators can use this case study in IS or IS project management classes as a resource for explaining pitfalls and complexities in the ISD process. Students could comprehensively analyze the case's main challenges and identify which best practices can minimize challenges in future ISD outsourcing projects. The following set of questions may help stimulate the students' thinking processes:

1) Why did the project—which seemed to progress well in the early stages—turn out so badly?
2) What best practices could have minimized the challenges that this ISD project faced?
3) What role did the various team members (e.g., business analysts, technical team, and client) have in minimizing the challenges that this ISD project faced?
4) If the vendor organization could identify another potential client for a similar project, how could the team members ensure that the project would succeed?

In the following teaching notes, we identify three learning objectives and outline the suggested teaching approach for the following topics: ISD outsourcing, ISD project processes, ISD process models, members of ISD project teams, modularization, interdependencies, volatile client requirements, and IS outsourcing maturity levels. We provide sample discussion questions and link the discussion topics to the literature and the relevant sections in the case study (Sections 7 and 8).

7 A Teaching Case on Information Systems Development Outsourcing: Lessons from a Failure

7.1 Teaching Objectives

1) To help students understand the importance of the requirement engineering stage in ISD outsourcing projects and the potentially serious consequences of requirement engineering issues.
2) To apprise students about the role that various team members (e.g., business analysts, software engineers) play in the requirement engineering process.
3) To explain ISD outsourcing challenges to students and to encourage them to factor in such challenges when planning ISD outsourcing projects.

8 Suggested Teaching Approach/Plan

8.1 ISD Outsourcing

The term “outsourcing” refers to using “external agents to perform an organizational activity” (King & Malhotra, 2000, p. 1). According to Statista (2019), the global outsourcing market reached about US$85.6 billion in 2018. Some leading IT outsourcing service providers include IBM, Accenture, Deloitte, and PwC (Gartner, 2017). In surveying industry professionals[7], Deloitte (2016) found that 72 percent of respondents’ organizations had outsourced their IT functions and the respondents expected a further 31 percent increase in outsourcing in the future (Deloitte, 2016).

ISD outsourcing remains one of the most popular types of outsourcing, and ISD outsourcing initiatives continue to see strong and continuous growth (Erickson-Harris, 2014; Nuwangi, Sedera, & Srivastava, 2018; Remus & Wiener, 2012; Willmott, 2012). According to Khan, Niazi, and Ahmad (2011), ISD outsourcing involves a contract-based relationship between client and vendor organizations wherein the client contracts out all or part of its ISD activities to the vendor. As Sakthivel (2007) highlights, most of Fortune 500 companies outsource their ISD activities to developing countries.

ISD outsourcing involves inherently complexity due to multiple, and often conflicting, client requirements, incongruence in interpreting client requirements, scope and definition creep, and the involvement of many user groups (Choudhury & Sabherwal, 2003; Dingsøyr, Nerur, Balijepally, & Moe, 2012). As such, ISD outsourcing projects notoriously do not provide the agreed deliverables on time (Nakatsu & Iacovou, 2009; Savolainen, Ahonen, & Richardson, 2012; Srivastava & Teo, 2012). According to the Standish Group (2014), on average, organizations complete only 16.2 percent of ISD outsourcing projects in the expected time and budget. Even more alarmingly, the Standish Group has estimated that organizations cancel 31.1 percent of ISD projects before completion (Standish Group, 2014), and Wojewod and Hastie (2015) found that 19 percent of projects completely fail.

8.1.1 Sample Discussion Questions

1) Using the case study as an example, discuss the main risks of ISD-outsourcing projects.

Key points to guide student discussions and answers: this question supports the third learning objective. Students should be able to discuss factors such as how vendors’ lack of understanding client requirements, requirement miscommunication, vendor organization losing control of project deadlines, and volatile requirements leading to project delays and failures. Students could also highlight the importance of the issues raised through the knowledge gaps between the client and vendor and why vendors do not accurately identify and document client requirements. They should be able to explain that the lack of business and technical knowledge among the team members could lead to project failures. We provide relevant information in Sections 3.4 and 4.

[7] More than 85 percent of the respondents came from companies with over US$1 billion annual revenue.

2) Using the case study as an example, discuss the importance of risk management in ISD outsourcing projects.

Key points to guide student discussions and answers: students could refer to Kliem (1999) and Aubert, Dussault, Patry, and Rivard (1999) to understand the importance of risk management and the risk management framework, respectively. Students should be able to highlight the importance of evaluating how much risk projects may involve and implementing relevant risk-management mechanisms. Students could identify that team members could have minimized the issues in the case study if they had identified the risks associated with requirement ambiguity, scope uncertainty, and interdependencies between modules and had applied appropriate mechanisms that targeted potentially high-risk elements. We provide relevant information in Section 4.

8.2 ISD Processes

ISD projects commence with a contractual agreement between the client and the ISD company. ISD projects comprise several stages such as requirement engineering, design, development, and testing (Boehm, 1988; Kneuper, 2018):

1) Requirement engineering stage: in this stage, business analysts identify and document the client requirements by conducting several discussions with the client. This stage has several substages: 1) requirement elicitation; 2) requirement analysis, modelling, and communication; 3) agreement on the requirements, and 4) evolution of requirements (Nuseibeh & Easterbrook, 2000). Vendors produce BRSs that describe the client requirements during the requirement engineering stage.
2) Design stage: the design stage involves conceptualizing and framing the final IS solution according to the requirements identified in the requirement analysis stage. The IS design includes architectural design and component and algorithm design. One produces several documents such as design specifications, interface design specifications, and test plans in this stage.
3) Development stage: the software engineering team conducts IS development during this stage. This stage includes writing, maintaining, and integrating the source code of the final IS solution. The software engineers need to develop the relevant requirement modules assigned to them.
4) Testing stage: this stage involves estimating the IS solution's quality. The QA team follows various testing approaches such as integration testing, load testing, and system testing to ensure that the software solution: 1) executes the functions accurately, 2) executes the functions in the expected time, and 3) meets the client requirements.

8.2.1 Sample Discussion Questions

1) Using the case study as an example, discuss the challenges in the four ISD project stages (i.e., requirement engineering, design, development, and testing).

Key points to guide student discussions and answers: students can explain that each ISD stage has its own challenges. Challenges in the requirement engineering stage and design stage include insufficient understanding of client requirements, lack of technical knowledge among consultants, lack of knowledge-transfer mechanisms, and lack of client commitment. The development stage can include challenges such as technical and functionality limitations and lack of communication between the consultant and technical teams. The testing stage may include challenges such as technical limitations in executing the tests and lack of business knowledge to accurately identify the test scenarios and test cases. We provide relevant information in Sections 3.4 and 4.

2) Explain how an issue in one stage could lead to issues in the other stages (e.g., how issues in the requirement engineering stage could lead to issues in the development stage).

Key points to guide student discussions and answers: this question focuses on triggering discussion related to the first learning objective. As we discuss in the case study, the vendor could not implement some initially agreed-on requirements due to interdependencies and clashes between requirement modules. During the testing stage, the QA team could not differentiate between the defects and accurate system-execution methods. Students should be able to identify that the lack of information in the BRSs constituted the root cause of these issues. They should be able to recognize that issues that originated during the requirement engineering phase led to issues in other stages. For example, because the vendor updated the BRSs frequently, the software engineers had to update the software code regularly. We provide relevant information in Section 4.4.

8.3 Information Systems Development Process Models

Organizations have used process models in ISD projects as a mechanism to helps elect the order in which they conduct the IS development stages (Boehm, 1988). Process models explain "what we do next" and "how long we continue to do it" (Boehm, 1988, p. 61). We can subdivide ISD process models into two types: plan driven and agile. Plan-driven process models suit projects in stable environments, and agile process models suit projects in volatile environments (Bose, 2008; Kneuper, 2018). Plan-driven process models include the waterfall model, V-model, and the rational unified process. Agile process models include scrum, Boehm's spiral model, and extreme programming (Kneuper, 2018).

8.3.1 Sample Discussion Questions

1) Discuss the differences between plan-driven process models and agile process models.

Key points to guide student discussions and answers: students could refer to Kneuper (2018) and Petersen and Wohlin (2010) who explain plan-driven and agile process models. Students can discuss that plan-driven process models suit projects in stable environments, while agile process models suit projects in volatile environments. Plan-driven process models require: 1) clear specifications of the expected outcomes of the project, 2) detailed explanations of the milestones, and 3) rigorous change requests after the implementation of the project (Petersen & Wohlin, 2010). Students should be able to identify that the agile process models emphasize self-regulated teams so that the teams can survive in volatile project environments. We provide relevant information in Section 3.2.

2) Which type of process model best suits the case study? Provide the rationale for your selection.

Key points to guide student discussions and answers: students could refer to Kneuper (2018) for explanations of process models. Students can explain that the case project followed the incremental and iterative development method whereby the vendor team members developed the system through a series of cycles. However, the team members could not complete the project successfully mainly because they miscommunicated with and did not sufficiently understand the client requirements. Students should be able to identify that, following Scrum, had the vendor adopted an agile approaches, it could have minimized the challenges in the project. Scrum increases transparency by using practices such as daily scrum meetings, sprint reviews, and product backlog artifact (Kneuper, 2018). During daily scrum meetings, team members (such as the business analysts, software engineers and QA team in the case project) can discuss the project's progress and clarify any doubts. During the sprint review, team members demonstrate software. The client can also join these sprint reviews to obtain regular updates (Permana, 2015). We provide relevant information in Section 3.2.

8.4 Members of ISD Project Teams

ISD projects comprise a team that focus on completing the IS solution according to the client requirements in the stipulated time and in the stipulated budget. An ISD project allocates team members to groups depending on their tasks and responsibilities (Figure 2 illustrates the team structure in the case project):

- **Group A** (project management team): the project management team prepares the project plans and ensure all the teams execute the project according to the project plans. The project management team includes project managers, junior project managers, associate project managers, and senior project managers.
- **Group B** (business analysis team): the business analysis team primarily writes the BRSs, which describe the client requirements. Thus, the business analysis team works as a conduit between the clients and the other team members in the ISD project. The business analysis team includes business analysts, senior business analysts, consultants, and senior consultants.
- **Group C** (technical team): the technical team writes the design specifications and develops the software according to the BRSs and design specifications. The technical team includes

software engineers (also known as developers), senior software engineers, specialist software engineers, principal software engineers, technical leads, and senior technical leads.

- **Group D** (QA team): the QA team writes the test scenarios, tests case specifications, and tests software according to the test scenarios and test case specifications. The QA team includes associate QA engineers, QA engineers, senior QA engineers, and QA managers.

8.4.1 Sample Discussion Questions

1) Using the case study as an example, explain how an ISD team could minimize the challenges in ISD outsourcing projects.
    - What responsibilities does the business analysis team have in minimizing these challenges?
    - What responsibilities does the technical team have in minimizing these challenges?
    - What responsibilities does the client have in minimizing these challenges?

Key points to guide student discussions and answers: this question focuses on the second learning objective. Students can identify the responsibilities of the team as follows:

- Business analysts: ensure that they accurately identify and document the client requirements and that they communicate them to other team members
- Technical team members: ensure they develop software solutions according to the client requirements and inform business analysts about the project's technical limitations
- Client: provide commitment to the project and clarify any issues regarding requirements.

2) Goh, Pan, and Zuo (2013) explain the different types of an ISD team's capabilities: technical capability, behavioral capability, business capability, and infrastructure capability. Discuss which types of capabilities the ISD team members in the case project need.

Key points to guide student discussions and answers: students could refer to Goh et al. (2013) for further clarification on the different capability types. Goh et al. (2013, p. 728) define technical capability as "the ability of the IT personnel based on their specific expertise in technical areas". Students should be able to highlight that the business analysts in the teaching case did not have sufficient technical capabilities to identify the system's technical limitations. As a result, they had to remove some technical capabilities when conducting the project. Students should be able to explain that the vendor could have minimized this issue if it allowed the technical team (i.e., the team members who had sufficient technical capabilities) to provide their feedback during the requirement engineering process. The teams did not sufficiently interact and collaborate during the project, which indicates that team members needed to develop their behavioral capabilities. The students could explain that the team members did not understand the overall business environment and the client's organizational context. Thus, the ISD team's business capabilities should be enhanced. We provide relevant information in Sections 3.4, 4.1, 4.2, 4.3, and 4.4.

8.5 Modularization

Modularization involves decomposing "complex tasks into simpler portions so they can be managed independently and yet operate together as a whole" (Mikkola & Skjøtt-Larsen, 2004, p. 354). Modularization in ISD outsourcing projects includes decomposing ISD tasks into simpler portions, which enables team members to manage the decomposed tasks independently (Nuwangi, 2016). When team members integrate the decomposed tasks, the tasks should operate together as a whole. According to Cataldo (2007), ISD outsourcing projects benefit through modularization by minimizing the interdependencies between outsourcing team members. Pine (1993) identifies six types of modularization: 1) component-sharing modularity, 2) component-swapping modularity, 3) cut-to-fit modularity, 4) mix modularity, 5) bus modularity, and 6) sectional modularity.

8.5.1 Sample Discussion Questions

1) Which type of modularization best suits the case project?

Key points to guide student discussions and answers: students could refer to Pine (1993) for further information on the six types of modularization. The project used the component-sharing modularity, which led to fluctuations in BRSs. Students should be able to identify that teams can use sectional modularity to minimize fluctuations. According to Pine (1993, p. 208), sectional modularity provides "the greatest degree of variety and customization.... [It] allows the configuration of any number of different types of components in arbitrary ways—as long as each component is connected to another at standard interfaces.". Students should be able to recommend that the project should have documented BRSs with sectional modularity (i.e., a separate BRS covered each module). When teams have to share a particular module with other modules, the BRSs for other modules should include only their particular inputs and outputs. A BRS for a module should only include all the information related to it (see Figure 6). Students should be able to explain that, when one documents BRSs following sectional modularity, a change to one BRS has less impact on the other modules. Thus, one can minimize fluctuations in BRSs. We provide relevant information in Section 4.4.

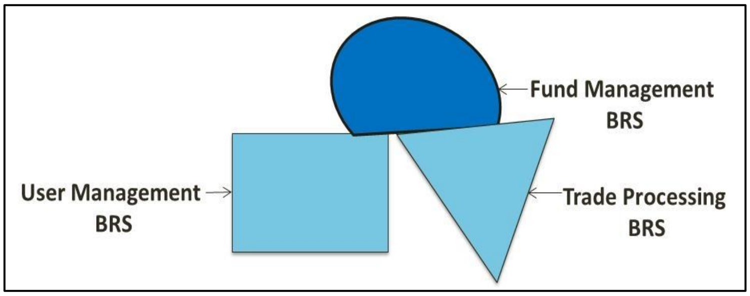


Figure 6. Sectional Modularity Example

2) Using the case study as an example, discuss the impact that modularization has on project controls.

Key points to guide student discussions and answers: students could refer to Tiwana (2008) for further information on the relationship between modularization and project controls. Students should be able to identify BRSs as a main control mechanism for team members. For example, the software engineers had to follow BRSs in their software development tasks. In documenting the BRSs, consultants followed the component-sharing modularity, which created fluctuations in the BRSs. As a result, the project had volatile expected outcomes, which led to project-management difficulties. Therefore, students should be able to explain the importance of considering modularization when deciding on the control mechanisms for projects. We provide relevant information in Section 4.4.

8.6 Interdependencies

Interdependencies between modules dictate how modules depend on each other (Gerdin, 2005). Many argue that interdependencies can have a direct impact on project performance (Caglio & Ditillo, 2012) and project success (Cataldo, Mockus, Roberts, & Herbsleb, 2009). According to Hoegl, Weinkauf, and Gemuenden (2004), the failure to manage task interdependencies between modules leads to unnecessary rework and duplication. When assessing interdependencies, one should use two important aspects: 1) coupling, and 2) cohesion (Nof, 2009). Coupling refers to the interdependencies of one module with other modules, and cohesion refers to a module's internal interdependencies (Kwong, Mu, Tang, & Luo, 2010).

8.6.1 Sample Discussion Question

1) Using the case study as an example, discuss the impact that coupling and cohesion have on project governance.

Key points to guide student discussions and answers: students should be able to identify that a change in one module can beget changes in other modules due to high coupling. These changes create a volatile environment for the project, which makes it difficult to govern. Students should be able to highlight the importance of minimal coupling between modules. Moreover, they should be able to discuss the importance of high cohesion in individual modules. We provide relevant information in Section 4.4.

8.7 Volatile Client Requirements

Changes to client requirement commonly occur in large ISD projects (Gefen, Wyss, & Lichtenstein, 2008; Yadav, Nath, Adya, & Sridhar, 2016). According to Wang, Ju, Jiang, and Klein (2008), when the client requirements change, the ISD team focuses on a moving target, which diminishes the IS solution's quality. Modifications to the software requirements may create the need to revise many parts of the software code (Ying, Murphy, Ng, & Chu-Carroll, 2004). Most importantly, changing the software code requires the ISD team to identify other related software code (Shirabad, Lethbridge, & Matwin, 2000). Maruping, Venkatesh, and Agarwal (2009) discuss using the agile methodology in ISD projects with volatile client requirements.

8.7.1 Sample Discussion Questions

1) What impact do volatile client requirements have on project governance? Use the case study as an example to support your arguments.

Key points to guide student discussions and answers: this question focuses on triggering discussion related to the second learning objective. Students could refer to Maruping et al. (2009) for further information on the relationship between volatile client requirements and project governance. Students should be able to explain that the project had volatile client requirements because 1) the client requested new requirements after it signed off the BRSs, 2) consultants removed some requirements due to interdependencies between modules, and 3) business analytics inaccurately identified the client requirements. The team members had to develop the software according to the client requirements, so updates in the client requirements created project-governance issues. Students should be able to explain the importance of accurately identifying client requirements and the importance of stable requirements in ISD projects. We present relevant information in Sections 4.1 and 4.4.

8.8 IS Outsourcing Maturity Levels

Fairchild (2004) describe the five levels of the outsourcing management maturity model as follows:

- Level 1 (vendor management fundamentals): this level features minimum levels of contract management processes, misaligned expectations, and lack of trust between client and vendor teams
- Level 2 (defined service outcome): a project can reach this level if the project has formal processes and benchmarks
- Level 3 (measurement): a project can reach this level if the project has service-level agreements and metrics
- Level 4 (trust): a project can reach this level when the outsourcing arrangements have improved service quality, reduced cost, and improved responsiveness
- Level 5 (recognized business value): a project can reach this level when the client and vendor achieve the outsourcing goals and project outcomes (e.g., performance, efficiency) exceed the service-level agreements.

8.8.1 Sample Discussion Questions

1) Using the outsourcing management maturity model, discuss the case project's maturity level. Provide the rationale for your answer.

Key points to guide student discussions and answers: students can refer to Fairchild (2004) for further information on maturity levels. The case project featured minimum levels of contract management processes, misaligned expectations, and lack of trust between client and vendor teams. Moreover, the client and vendor did not have well-established management fundamentals such as process ownership, cost containment, and project management. Students should be able to understand that the maturity level of this project reached the first level (vendor management fundamentals). We provide relevant information in Sections 3.4, 4.1, 4.2, 4.3, 4.4 and 4.5.

9 Suggested Additional Readings

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer Modelling and Simulation of Smart and Green Computing Systems*, *21*(5), 61-72.

Kneuper, R. (2018). *Software processes and life cycle models: An introduction to modelling, using and managing Agile, plan-driven and hybrid processes.* New York, NY: Springer.

Pine, B. J. (1993). *Mass customization the new frontier in business competition.* Boston, MA: Harvard Business School Press.

Cataldo, M. (2007). *Dependencies in geographically distributed software development: Overcoming the limits of modularity* (doctoral dissertation). Carnegie Mellon University, Pittsburgh, PA.

Tiwana, A. (2008). Does technological modularity substitute for control? A study of alliance performance in software outsourcing. *Strategic Management Journal*, *29*(7), 769-780.

Goh, J. C.-L., Pan, S. L., & Zuo, M. (2013). Developing the agile IS development practices in large-scale IT projects: The trust-mediated organizational controls and IT project team capabilities perspectives. *Journal of the Association for Information Systems*, *14*(12), 722-756.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A control theory perspective on agile methodology use and changing user requirements. *Information Systems Research*, *20*(3), 377-399.

References

Aubert, B. A., Dussault, S., Patry, M., & Rivard, S. (1999). Managing the risk of IT outsourcing. In *Proceedings of the Hawaii International Conference on Systems Sciences.*

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer Modelling and Simulation of Smart and Green Computing Systems, 21*(5), 61-72.

Bose, I. (2008). Lessons learned from distributed agile software projects: A case-based analysis. *Communications of the Association for Information Systems, 23*, 619-632.

Caglio, A., & Ditillo, A. (2012). Opening the black box of management accounting information exchanges in buyer–supplier relationships. *Management Accounting Research, 23*(2), 61-78.

Cataldo, M. (2007). *Dependencies in geographically distributed software development: Overcoming the limits of modularity* (doctoral dissertation). Carnegie Mellon University, Pittsburgh, PA.

Cataldo, M., Mockus, A., Roberts, J. A., & Herbsleb, J. D. (2009). Software dependencies, work dependencies, and their impact on failures. *IEEE Transactions on Software Engineering, 35*(6), 864-878.

Choudhury, V., & Sabherwal, R. (2003). Portfolios of control in outsourced software development projects. *Information Systems Research, 14*(3), 291-314.

Chua, A., & Lam, W. (2005). Why KM projects fail: A multi-case analysis. *Journal of Knowledge Management, 9*(3), 6-17.

Deloitte. (2016). *Deloitte's 2016 global outsourcing survey.* Retrieved from https://www2.deloitte.com/content/dam/Deloitte/nl/Documents/operations/deloitte-nl-s&o-global-outsourcing-survey.pdf

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

Erickson-Harris, L. (2014). IT outsourcing spend: Slow and steady for 2014. *Nearshore Americas*. Retrieved from http://www.nearshoreamericas.com/outsourcing-spend-slow-steady-2014/

Fairchild, A. M. (2004). Information technology outsourcing (ITO) governance: An examination of the outsourcing management maturity model. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences.*

Gartner. (2017). *Market share analysis: IT outsourcing services, worldwide, 2016.* Retrieved from https://www.gartner.com/doc/3744417/market-share-analysis-it-outsourcing

Gefen, D., Wyss, S., & Lichtenstein, Y. (2008). Business familiarity as risk mitigation in software development outsourcing contracts. *MIS Quarterly, 32*(3), 531-542.

Gerdin, J. (2005). The Impact of departmental interdependencies and management accounting system use on subunit performance. *European Accounting Review, 14*(2), 297-327.

Goh, J. C.-L., Pan, S. L., & Zuo, M. (2013). Developing the agile IS development practices in large-scale IT projects: The trust-mediated organizational controls and IT project team capabilities Perspectives. *Journal of the Association for Information Systems, 14*(12), 722-756.

Hoegl, M., Weinkauf, K., & Gemuenden, H. G. (2004). Interteam coordination, project commitment, and teamwork in multiteam R&D projects: A longitudinal study. *Organization Science, 15*(1), 38-55.

Khan, S. U., Niazi, M., & Ahmad, R. (2011). Barriers in the selection of offshore software development outsourcing vendors: An exploratory study using a systematic literature review. *Information and Software Technology, 53*(7), 693-706.

King, W. R., & Malhotra, Y. (2000). Developing a framework for analyzing IS sourcing. *Information & Management, 37*(6), 323-334.

Kliem, R. L. (1999). Managing the risks of outsourcing agreements. *IS Management, 16*(3), 91-93.

Kneuper, R. (2018). *Software processes and life cycle models: An introduction to modelling, using and managing Agile, plan-driven and hybrid processes*. New York, NY: Springer.

Kwong, C. K., Mu, L. F., Tang, J. F., & Luo, X. G. (2010). Optimization of software components selection for component-based software system development. *Computers & Industrial Engineering, 58*(4), 618-624.

Lyytinen, K. & Robey, D. (1999). Learning failure in information systems development. *Information Systems Journal,* 9(2), 85-101.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A control theory perspective on agile methodology use and changing user requirements. *Information Systems Research, 20*(3), 377-399.

Mikkola, J. H., & Skjøtt-Larsen, T. (2004). Supply-chain integration: Implications for mass customization, modularization and postponement strategies. *Production Planning & Control, 15*(4), 352-361.

Nakatsu, R. T., & Iacovou, C. L. (2009). A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel delphi study. *Information & Management, 46*(1), 57-68.

Nof, S. Y. (Ed.). (2009). *Handbook of automation*. New York, NY: Springer.

Nuseibeh, B., & Easterbrook, S. (2000). Requirements engineering: A roadmap. In *Proceedings of the Conference on the Future of Software Engineering.*

Nuwangi, S. M. (2016). *The impact of modularisation on information system development outsourcing project control* (doctoral thesis). Queensland University of Technology, Australia.

Nuwangi, S. M., Sedera, D., & Srivastava, S. C., (2018). Multi-layered control mechanisms in software development outsourcing. In *Proceedings of the Pacific Asia Conference on Information Systems.*

Pine, B. J. (1993). *Mass customization the new frontier in business competition*. Boston, MA: Harvard Business School Press.

Permana, P. A. G. (2015). Scrum method implementation in a software development project management. *International Journal of Advanced Computer Science and Applications, 6*(9), 198-204.

Petersen, K., & Wohlin, C. (2010). The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering, 15*(6), 654-693.

Remus, U., & Wiener, M. (2012). The amount of control in offshore software development projects. *Journal of Global Information Management, 20*(4), 1-26.

Sakthivel, S. (2007). Managing risk in offshore systems development. *Communications of the ACM, 50*(4), 69-75.

Savolainen, P., Ahonen, J. J., & Richardson, I. (2012). Software development project success and failure from the supplier's perspective: A systematic literature review. *International Journal of Project Management*, 30(4), 458-469.

Senarath, S., & Copp, R. (2015). Credit default swaps and the global financial crisis: Reframing credit default swaps as quasi-insurance. *Global Economy and Finance Journal*, *8*(1), 135-149.

Senarath, S. (2017). The Dodd-Frank Act doesn't solve the principal-agent problem in asset securitisation. *LSE Business Review.* Retrieved from https://blogs.lse.ac.uk/businessreview/2017/04/21/the-dodd-frank-act-doesnt-solve-the-principal-agent-problem-in-asset-securitisation/

Shirabad, J. S., Lethbridge, T. C., & Matwin, S. (2000). Supporting maintenance of legacy software with data mining techniques. In *Proceedings of the Conference of the Centre for Advanced Studies on Collaborative Research.*

Srivastava, S. C., & Teo, T. S. H. (2012). Contract performance in offshore systems development: Role of control mechanisms. *Journal of Management Information Systems, 29*(1), 115-158.

Standish Group. (2014). *The Standish Group report: CHAOS*. Retrieved from https://www.projectsmart.co.uk/white-papers/chaos-report.pdf

Statista. (2019). *Global market size of outsourced services from 2000 to 2018 (in billion U.S. dollars)*. Retrieved from https://www.statista.com/statistics/189788/global-outsourcing-market-size/

Tiwana, A. (2008). Does technological modularity substitute for control? A study of alliance performance in software outsourcing. *Strategic Management Journal, 29*(7), 769-780.

Wang, E. T. G., Ju, P.-H., Jiang, J. J., & Klein, G. (2008). The effects of change control and management review on software flexibility and project performance. *Information & Management, 45*(7), 438-443.

Wojewod, S., & Hastie, S. (2015). Standish Group 2015 chaos report. *InfoQ.* Retrieved from http://www.infoq.com/articles/standish-chaos-2015

Willmott, D. (2012). Report: CIOs to accelerate outsourcing in 2013. *Dice*. Retrieved from https://insights.dice.com/2012/08/29/outsourcing-accelerates-2013/

Yadav, V., Nath, D., Adya, M., & Sridhar, V. (2016). Considerations for effective requirements analysis in offshore software development projects: Lessons from multi-method research. *Communications of the Association for Information Systems, 39*, 188-213.

Ying, A. T. T., Murphy, G. C., Ng, R., & Chu-Carroll, M. C. (2004). Predicting source code changes by mining change history. *IEEE Transactions on Software Engineering, 30*(9), 574-586.

Appendix: Revision History of Fund Processing BRS

Table A1. Revision History of Fund Processing BRS

Date	Version	Description
5 December, 2011	1.03_4	Account Creation [Amended] “Account Structure” as “Account Creation” [Removed] Separate account structures are maintained for Cash and Delivery accounts. [Added] XXXXXXXXXXX[8] [Removed] Account category since the system will maintain two categories for cash and delivery separately. [Added] XXXXXXXXXXX [Added] XXXXXXXXXXX [Removed] XXXXXXXXXXX [Added] Reference number [Removed] Reference number in a ledger account. [Added] XXXXXXXXXXX [Added] XXXXXXXXXXX [Amended] XXXXXXXXXXX [Added] XXXXXXXXXXX [Removed] Each entity created in the system will have an accounting structure attached to it. [Removed] XXXXXXXXXXX [Removed] The accounting structure will dictate the accounts created by the system for each instance [Added] XXXXXXXXXXX [Removed] XXXXXXXXXXX [Removed] The levels in the accounting structure since accounting structures will not be maintained in the system. [Removed] Example for maintaining accounts at multiple levels [Removed] A default accounting structure will be configured for each entity in the system. [Removed] XXXXXXXXXXX

[8] We conceal sensitive information in the table in order to maintain confidentiality.

About the Authors

Subasinghage Maduka Nuwangi is a lecturer at Auckland University of Technology, New Zealand. She completed her PhD studies at the Queensland University of Technology, Australia. Her research interests include software development, modularization, outsourcing and knowledge management. Her research has been published in *VINE: The Journal of Information and Knowledge Management Systems* and many conference proceedings (e.g., International Conference on Information Systems, Pacific Asia Conference on Information Systems, European Conference on Information Systems, Australasian Conference on Information Systems, and Australian & New Zealand Academy of Management Conference).

Darshana Sedera is a Professor at Swinburne Business School, Swinburne University of Technology in Melbourne, Australia. He has published over 180 publications in major refereed journals and conferences. His publications have appeared in *Journal of the Association for Information Systems*, *Journal of Strategic Information Systems*, *Information & Management*, Infor*mation Technology & People*, and *Communications of the Association for Information Systems*.