Case Study on Re-Architecting of Established Enterprise Software Product: Major Challenges Encountered and SDM Prescriptions from Lessons Learned

By
Kok-Seng (Darrel) Quah

Submitted to the System Design and Management Program in Partial Fulfillment of the Requirements for the Degree of Master of Science in Engineering and Management at the Massachusetts Institute of Technology

May 2005

© 2005 Kok-Seng (Darrel) Quah
All rights reserved

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part.

Signature of Author

Kok-Seng (Darrel) Quah
System Design and Management Program
May 2005

Certified by

Daniel Whitney
Thesis Supervisor
Center for Technology, Policy & Development

Massachusetts Institute of Technology
MAY 27 2005
Barker Libraries
Case Study on Re-Architecting of
Established Enterprise Software Product:
Major Challenges Encountered and SDM
Prescriptions from Lessons Learned

By
Kok-Seng (Darrel) Quah

Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management
at the
Massachusetts Institute of Technology
May 2005

Abstract

The paper studies a real word project of an enterprise software product re-architecting at a mid-sized telecommunication company. It begins with a description of the company and the software product, as well as an elaboration of the project under study. Using written surveys and follow-up interviews as the primary data gathering tools, the paper collects and tabulates first-hand experience and opinions from key project participants. Based on the survey results, the paper proposes an integrative implementation framework, based primarily on literature reviews in offshore outsourcing, systems and project management (SPM) and product design and development (PDD), for a detailed analysis of key challenges encountered by the project under study. The paper also investigates if specific key challenges could have been managed or influenced by the application of specific methods and tools within the proposed framework.

Thesis Supervisor:
Daniel E. Whitney, PhD
Senior Research Scientist, Center for Technology, Policy & Development
This page is intentionally left blank
Acknowledgements

The study was made possible with the help of many people. First and foremost, I would like to thank my SDM thesis advisor, Professor Daniel Whitney, for agreeing to supervise my work. His guidance fundamentally shaped the direction and outcome of this study.

The initial concept of the study started as a term project for ESD.36J Systems and Project Management in Fall Semester 2004. My project teammates, Tony Zhang and Henry Chen, were instrumental in revising the original survey questionnaire and interpreting the written survey responses. Their excellent work on the term project laid some of the groundwork for the study.

I would like to thank my former colleagues for sharing their experiences in the project under study. I want to express my gratitude especially to those who agreed to participate in the survey and did so with dedication and candor. If I were not constrained by the confidentiality of the study, I would have publicly acknowledged each of them here.

Finally, I want to thank a few of my peers - Mal Atherton and Tim Flynn - for being my great teammates in countless term projects throughout the SDM program. In addition, I want to express my gratitude to a few great peers - Dan Dombak, Hide Moroi, and Peter Manickas - for giving me much of the support I needed for pursuing my studies at MIT.

Thank you all!

May 3, 2005
This page is intentionally left blank
# Table of Contents:

1 **CHAPTER 1: INTRODUCTION** ..................................................................................11
   1.1 BACKGROUND ..................................................................................................11
   1.2 MOTIVATION ..................................................................................................11
   1.3 THESIS STATEMENT AND PRIMARY RESEARCH OBJECTIVES ..................12
   1.4 THESIS CONTENT ..........................................................................................13

2 **CHAPTER 2: THE CASE STORY – ENTERPRISE SOFTWARE PRODUCT RE-ARCHITECTING** .................................................................17
   2.1 CHAPTER INTRODUCTION .............................................................................17
   2.2 COMPANY PROFILE .....................................................................................17
   2.3 PRODUCT PROFILE – POINT-OF-SALE (POS) SYSTEM ................................19
      2.3.1 Product Architecture – First Generation POS System .......................22
   2.4 CASE: PRODUCT RE-ARCHITECTING PROJECT ........................................25
      2.4.1 Project Initiation .....................................................................................25
      2.4.2 Project Architecture – Second Generation POS System .......................27
      2.4.3 Project Planning ......................................................................................29
      2.4.4 Project Implementation ..........................................................................35
      2.4.5 Project Outcome .....................................................................................39
   2.5 CHAPTER SUMMARY .....................................................................................39

3 **CHAPTER 3: SURVEY IMPLEMENTATION AND FINDINGS** ..............................41
   3.1 CHAPTER INTRODUCTION .............................................................................41
   3.2 CASE OVERVIEW ..........................................................................................41
   3.3 SURVEY IMPLEMENTATION .........................................................................42
      3.3.1 Survey Targets .......................................................................................42
      3.3.2 Survey Focus ..........................................................................................44
      3.3.3 Survey Questionnaire ............................................................................44
   3.4 SURVEY FINDINGS AND FOLLOW-UPS .....................................................45
      3.4.1 Findings on Responses to Survey Questionnaire .....................................45
      3.4.2 Findings on Design Setbacks ..................................................................50
   3.5 CHAPTER SUMMARY .....................................................................................54

4 **CHAPTER 4: INTEGRATIVE IMPLEMENTATION FRAMEWORK** .......................55
   4.1 CHAPTER INTRODUCTION .............................................................................55
   4.2 CONTEXT SPECIFICITY ..................................................................................55
      4.2.1 Offshore Outsourcing – Overview .............................................................57
      4.2.2 Offshore Outsourcing – Trends and Statistics ...........................................58
   4.3 IMPLEMENTATION FRAMEWORK ..................................................................61
      4.3.1 Corporate Strategy and Business Decision Making ...................................63
      4.3.2 Organization Structure & Processes .......................................................70
List of Figures:

FIGURE 1 - GENERAL PRODUCT ARCHITECTURE OF POS SYSTEM ............................................................ 20
FIGURE 2 - HIGH LEVEL CLIENT-SERVER ARCHITECTURAL VIEW (PHYSICAL VIEW) .......................... 24
FIGURE 3 - RMS 4.X HIGH LEVEL THREE-TIER ARCHITECTURAL VIEW (LOGICAL VIEW) ............... 28
FIGURE 4 - PLANNED PRODUCT ROAD MAP ......................................................................................... 32
FIGURE 5 - HIGH-LEVEL MASTER PROJECT TIMELINE ....................................................................... 34
FIGURE 6 - OVERLAPPING OF TASKS .................................................................................................. 34
FIGURE 7 - PROJECT TEAM ORGANIZATIONAL STRUCTURE .............................................................. 36
FIGURE 8 - ACTUAL OUTCOME OF PRODUCT ROAD MAP ................................................................. 37
FIGURE 9 - GENERAL SURVEY QUESTIONNAIRE ................................................................................ 45
FIGURE 10 - TABULATION OF SURVEY RESPONSES ....................................................................... 46
FIGURE 11 - MAJOR PROJECT SETBACKS AND RAMIFICATIONS .................................................... 51
FIGURE 12 - OFFSHORE GROWTH: GARTNER’S PROJECTIONS FOR OFFSHORE OUTSOURCING ........... 59
FIGURE 13 - OUTSOURCING WORRIES AMONG US COMPANIES ................................................ 60
FIGURE 14 - IMPLEMENTATION FRAMEWORK FOR ENTERPRISE SOFTWARE RE-ARCHITECTING .... 62
FIGURE 15 - THE SKILLS NEEDED IF ONE IS DEPENDENT FOR KNOWLEDGE COMPARED TO THOSE NEEDED IF ONE IS DEPENDENT FOR CAPACITY ...................................................... 65
FIGURE 16 - MATRIX OF DEPENDENCY AND OUTSOURCING ........................................................ 66
FIGURE 17 - DIAGNOSTIC FRAMEWORK: OFFSHORE OUTSOURCING OF APPLICATION DEVELOPMENT .................................................................................................................. 69
FIGURE 18 - DECOMPOSITION OF RISK MANAGEMENT ..................................................................... 76
FIGURE 19 - FEASIBILITY OF POS 4.X FOR OFFSHORE OUTSOURCING ........................................ 84
FIGURE 20 - ORGANIZATION DESIGN FOR EFFECTIVE COMMUNICATION ...................................... 87

List of Tables:

TABLE 1 - DECISION DRIVING FACTORS FOR PROJECT INITIATION .................................................. 26
TABLE 2 - KEY PROJECT ATTRIBUTES DEFINED BY INTERNAL PROJECT SPONSOR ......................... 30
TABLE 3 - HIGH-LEVEL MASTER PROJECT PLAN .................................................................................. 33
TABLE 4 - ROLE OF SURVEY TARGETS ................................................................................................. 43
TABLE 5 - THREE FUNDAMENTAL SPM STRATEGIES ....................................................................... 73
TABLE 6 - SUMMARY OF PDD (SOFTWARE ENGINEERING) RISKS ................................................ 79
TABLE 7 - KEY ISSUES IDENTIFIED ON PROJECT UNDER STUDY ................................................... 82
TABLE 8 - ROLE OF SURVEY RESPONDENTS ...................................................................................... 107
This page is intentionally left blank.
1 Chapter 1: Introduction

1.1 Background

The author was a key team member of an engineering project on the re-architecting of an established enterprise software product at a mid-sized company. The author was primarily responsible for product scalability, integration, and configuration. The company initiated the proposal on product re-architecting with a key customer’s agreeing to be the primary project sponsor. It was the very first project that the company engaged an offshore outsourcing vendor in its product development efforts. The vendor was hired to partake in the design and development of a key product component. The main reasons behind the offshore outsourcing arrangement were to achieve lower costs and faster time to market than the company were able to if it had chosen to implement all product development efforts in house. Contrary to achieving lower costs and faster time to market, the project encountered numerous project setbacks, resulting in significant schedule slippage and cost overrun. Originally scheduled to complete in ten months, the project continued into its twenty-third month before it was halted by the primary project sponsor.

1.2 Motivation

The author is motivated to study the project carefully and investigate major challenges encountered throughout the product development life cycle. Among the areas of particularly strong interests to the author pertain to both the engineering and management aspects of the project, namely the project initiation, project management, and product design and development. The author intends to investigate if others in the enterprise software industry
have encountered and dealt with similar experiences. Furthermore, the author seeks to develop a framework for the analysis of the project and prescribe solutions for other projects that possess similar characteristics.

### 1.3 Thesis Statement and Primary Research Objectives

The case study entails a re-architecture of an established software product at a mid-size telecommunication company. The case warrants an in-depth study due to its exhibition of many classical project symptoms despite the constancy of functional requirements, where are already fully defined in the existing product architecture. The appeal of the case is further enhanced by the implementation of a new technology by the company involving its very first attempt to engage an offshore outsourcing provider in product design and development efforts. The constancy of functional requirements is generally perceived as a strongly positive project attribute as it would preclude adverse impacts frequently created by unplanned requirement changes. Nonetheless, the case encountered a litany of woes related to both project management and product design and development. Among the issues it experienced were significant schedule slippage, cost overrun, and dissatisfactory product quality. The final product was never deployed in production.

The thesis entails a single case study applying SDM theories, methods, and tools. The primary research objectives consist of the following:

- Study the history and nature of a software re-architecting project under investigation.
- Investigate (via literature search) if recommended practice was followed by the project as well as the sufficiency of recommended practice.
- Identify and analyze key challenges faced by the project.
- Research and evaluate if key challenges can be managed or directly influenced by specific System Design and Management (SDM) theories, frameworks, methods and tools.

- Prescribe possible solutions for future projects of similar characteristics.

In addition, the thesis seeks to answer the following questions:

1. Given the "low-risk" nature of the project due to the constancy of product specification, why did the project still manage to encounter countless SPM and PDD issues?
2. What were the root causes that gave rise to these SPM and PDD issues?
3. Could the root causes be managed or influenced by specific methods and tools?
4. Specifically, which PDD and SPM framework and methods/tools can be applied to analyze and resolve problems facing the project under study?
5. Are there any case studies about software project failure to which the thesis can refer, particularly those failures pertaining to software re-architecting?

1.4 Thesis Content

The thesis is organized into three major chapters and a conclusion. Below is the caption of the chapter introduction from each major chapter:

**Chapter 2: The Case Story – Enterprise Software Product Re-Architecting**

The chapter intends to tell a complete story of an engineering project that involved re-architecting of an established enterprise software product. The project took place at a mid-size telecommunication company between 2001 and 2003. The chapter comprises two major sections. The first section describes the corporate history of the company and the evolution of
a product of the company, a point-of-sale (POS) system, starting from its inception to becoming a major product before it was specifically targeted for product re-architecting. The second section describes the entire project lifecycle involving the re-architecting of the POS system, starting with the project initiation and ending at the project outcome. The importance of the chapter lies in its serving as a foundation for a case study on the re-architecting of an established enterprise software product.

Chapter 3: Survey Implementation and Findings

The chapter is provisioned for the description of the survey implementation and the documentation of the survey findings. Starting with a brief overview of the case, it describes the survey implementation in terms of its survey targets, focus, and questionnaire. All responses to the survey questionnaire and the interview note with the program manager of the project under study are placed in Appendix B. Following the coverage on the survey implementation are the survey findings from the responses to survey questionnaire and the additional findings from survey follow-ups via email and telephone conversations with survey respondents. These survey findings form the basis for a detailed analysis of the project under study in the next chapter.

Chapter 4: Integrative Implementation Framework

This chapter intends to create an integrative analytical approach in a form of an implementation framework for analyzing the project under study. The implementation framework is integrative because it comprises both general management areas and engineering areas. Offshore outsourcing of application development is recognized as an
important factor in establishing context specificity for the implementation framework. Taking the specific context into account, the chapter creates an implementation framework specifically for software product re-architecting projects engaging offshore outsourcing vendors in the application development efforts. The implementation framework is applied to the analysis of key issues of the project under study tabulated from the survey findings in Chapter 3. The overall approach is highly constructive because it provides an integrative way to analyze the project study and establishes recommendations that are applicable to the re-architecting of enterprise software products within the aforementioned context specificity.
This page is intentionally left blank.
Chapter 2: The Case Story – Enterprise Software Product Re-Architecting

2.1 Chapter Introduction

The chapter intends to tell a complete story of an engineering project that involved re-architecting of an established enterprise software product. The project took place at a mid-size telecommunication company between 2001 and 2003. The chapter comprises two major sections. The first section describes the corporate history of the company and the evolution of a product of the company, a point-of-sale (POS) system, starting from its inception to becoming a major product before it was specifically targeted for product re-architecting. The second section describes the entire project lifecycle involving the re-architecting of the POS system, starting with the project initiation and ending at the project outcome. The importance of the chapter lies in its serving as a foundation for a case study on the re-architecting of an established enterprise software product.

2.2 Company Profile

The company under study, GBTL, was founded in 1989 to offer customer acquisition services to wireless service providers. The initial business focus of the company was leveraging information technology to provide enhanced transaction processing capability to handle what was then a tedious, manual process involved in acquiring customers for wireless telecommunication services. During the period when the project under study was initiated in December 2001, the annual revenue of the company was about $175 million. Slightly over
one fifth of its total revenue was contributed by a major wireless service provider for whom the project under study was provisioned.

GBTL started out by identifying and focusing on solving highly labor-intensive customer acquisition process in the wireless telecommunication industry. The customer acquisition process took days to complete as it involved the manual gathering, transmission and evaluation of customer application information as well as the manual activation of wireless telephone service on chosen wireless devices. When wireless telephone service was first introduced, applicants usually had to wait for days to obtain the status of their applications after they filled out the application form in person at a physical store of a wireless provider. Each paper-based application would subsequently undergo a lengthy evaluation process where the credit worthiness of the applicant would be retrieved and manually assessed. The approved applications would subsequently require the manual activation of a cellular phone selected by each subscriber. In a quick glance, the customer acquisition process was highly labor intensive and time consuming.

GBTL derived its success from its transaction processing capability that enabled providers of wireless services to manage customer transaction for greater profitability. Greater profitability was achieved by overcoming the highly labor intensive nature of the customer acquisition process. Using its proprietary information technology, the company enabled its clients to process customer applications and activate service subscriptions on a real time basis. The application process of each telecommunication client was painstakingly mapped out and streamlined by the company’s consulting services. Accordingly, the transaction
processing capability was customized based upon each telecommunication client’s distinct process flow.

The transaction processing capability of GBTL is distributed to its wireless telecommunication clients on the basis of a subscription business model. Under such a business model, the company developed and maintained the capability in-house and only a minimal amount of software in the areas of system interfaces was distributed to its clients. The pricing is based on subscription fees determined by multi-year business contracts and additional fees based on transaction volume. The deployment of the transaction processing capability requires only specification of system interfaces between the IT infrastructures of the company and its client. The specification of system interfaces determines the data format and exchange between the two IT infrastructures. Building on its initial success, GBTL expanded its core product lines to fraud prevention, credit qualification, payment processing, billing, and enhanced voice and data processing. The core product capabilities of the company reside in its intelligent, automated systems and human expertise to enable its wireless telecommunication clients to optimize the lifetime value of their customers*.

2.3 Product Profile – Point-of-Sale (POS) System

Point-of-Sale (POS) system can be generally described as specialized hardware equipment and software for retail business operations. It comprises specialized software, cash drawer, barcode scanner, and receipt printer. The POS system under study is a software application

* Lifetime Value is generally defined as the net profit a customer contributes to a business over the active life of the customer. In other words, it is “the total estimated monetary worth of a customer to a business over the course of the relationship”
(Source: http://ace.acadiau.ca/fps/business/FOLLOWS/BUSI4913/Glossary.htm)
used primarily in retail business units for processing the sales of wireless devices and services, and for managing the inventory of wireless devices and accessories. Wireless devices and accessories refer to cellular telephones and related accessories, such as battery chargers and cases. Wireless services consist primarily of telephone and pager services with arrays of service features and contracts. As illustrated in Figure 1, POS system is normally deployed throughout an enterprise. POS data are stored in database servers typically located in a secured data center. The system utilizes the internet as the medium for data exchange among various business units. The system is used for processing retail transactions at store locations and for enabling inventory management at distribution centers. At headquarters, the system provides retail data for business planning and monitoring purposes.
The POS system under study was not a part of GBTL’s initial core product offering, namely the transaction processing of customer acquisitions and service activations. The prevalence of wireless telephone usage in the United States started to take place in the mid-1990s. As demand accelerated, hundreds of wireless service providers, ranging from small, local resellers to regional providers to national providers, were established throughout the country. During this period of rapid growth, GBTL became very successful in acquiring wireless service providers as customers for its core product offering. By 1997 GBTL had become the provider of such transaction processing for multiple wireless service providers with strong national presence. The company added the POS system to its product offering after numerous requests from its existing and target customers. The company’s business strategy for revenue growth called for the integration of the POS system with its core transaction processing capability for meeting the needs of newly established wireless service providers.

In 1998 the POS system became a key product of the company when it was included in a comprehensive product offering to one of the top five providers of wireless service in the country. The POS system reached a critical milestone where it became an important product component of the company’s first end-to-end business solution for wireless retail operations. The product became strategically crucial as it enabled the company to secure a multi-year, multi-million dollar business contract that included the subscription to GBTL’s transaction processing capability. The business deal generated revenue that was in a high single-digit million dollars for the initial deployment of the POS system in 1998.
2.3.1 Product Architecture – First Generation POS System

The software application is a client/server system*. The client component is developed using the PowerBuilder programming language for desktop deployment running on Microsoft Operating Systems. The server component consists of business logics written in Oracle’s PL/SQL and business rules configured using Oracle’s Relational Database Management System (RDBMS). The processing of data processing tasks is distributed between the two components. For wireless service providers with high transaction volumes, the server component is optimized for multi-processor high-end server machines running on UNIX operating systems.

From the system deployment standpoint, the physical system of the POS system consists of two major components (Figure 2):

i. Consoles (Clients)

    **Store Operations:** The console for store operations (a.k.a. FrontOffice) consists of GUI forms provisioned for the entry, retrieval, and update of retail transactions. The GUI forms are designed according to the needs of the wireless provider’s business processes. The console is designed for deployment on a desktop located at a physical store location, and it interfaces with a set of specified peripheral devices. The consoles are connected to a central storage system and can be geographically dispersed.

---

* Client/Server system can be viewed as a networked information processing architecture in which application programs and information are stored on both clients and servers. Typically the system consists of multiple clients and a single server. The processing of tasks can be local on the clients, shared between a client and the server, or centralized on the server. This type of architecture became dominant in the development of data processing applications in the 1980s and early 1990s. 
(Source: [www.commnett.com/features/glossary.htm](http://www.commnett.com/features/glossary.htm))
Inventory Management: The console for inventory management (a.k.a. the BackOffice) also consists of GUI of POS designed for deployment on desktop. It is provisioned for the management of inventory throughout an enterprise, including individual stores, local warehouses, and distribution centers. The console is connected to a central data storage system.

ii. Data Storage (Server)

A database server is used by the wireless provider to store all data associated with POS transactions enterprise-wide. The server is a centralized component that resides in a secured data center facility. POS software execution taking place in the consoles send data to and retrieve data from the central data storage system. All retail transactions are recorded in real time and processed for financial and accounting reporting purposes in the end of each business day. The recorded data include product information and customer information.

The following system components are outside the POS system boundary but need to be included to a comprehensive understanding of the system:

i. POS Peripheral Devices:

Specific peripheral devices are needed for the operating functions at each store location but are not parts of the POS system as Figure 2. The peripheral devices for a client console comprise a cash drawer, receipt printer, barcode scanner and signature capture device.

ii. External Systems:
External systems provide additional functions not within the POS system boundary. The additional functions are core transaction processing capabilities, such as credit validation and service activation. Credit validation evaluates the credit worthiness of a subscription applicant and is performed by accessing credit-related databases and computing credit scores. Service activation turns on services on a specified wireless device and is performed by sending activation requests to a wireless service provider's backend infrastructure.

Figure 2 - High Level Client-Server Architectural View (Physical View)
2.4 Case: Product Re-Architecting Project

This section contains the description for the project under study. The case will serve as a function for the analysis of the project under study using System Design and Management (SDM) methods and tool. It attempts to describe the entire project lifecycle using primary sources of the company under study, a formal interview with Program Manager responsible for the project, and numerous informal conversations with key principals in the project team. The section covers the initiation, implementation, and final outcome of the project under study, and provides a high-level overview of the re-architected product architecture.

2.4.1 Project Initiation

GBTL's decision to re-architect the established enterprise software product, namely the POS system, was driven by three major factors in the marketplace. The three major factors consist of one external driver and two internal drivers (Table 1). The external driver comprised the existing customers of the POS system. The internal drivers consisted of the product development group and the product marketing and sales groups. All the aforementioned decision driving factors converged in 2001 and resulted in the initiation of the project under study.

<table>
<thead>
<tr>
<th>External Driver</th>
<th>Internal Driver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customers</td>
<td>The existing customers requested an open architecture as they continued to integrate the POS system into its IT infrastructure.</td>
</tr>
<tr>
<td>Internal Driver</td>
<td></td>
</tr>
</tbody>
</table>
The marketing group needed a new technology to strengthen the product positioning of the POS system.

The sales group required that the product be developed with new system architecture for the acquisition of new clients.

The engineering group needed the product to possess better customization capability in terms of development cost and time.

<table>
<thead>
<tr>
<th>Product Marketing and Sales</th>
<th>The marketing group needed a new technology to strengthen the product positioning of the POS system.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The sales group required that the product be developed with new system architecture for the acquisition of new clients.</td>
</tr>
<tr>
<td>Product Development</td>
<td>The engineering group needed the product to possess better customization capability in terms of development cost and time.</td>
</tr>
</tbody>
</table>

Table 1 - Decision Driving Factors for Project Initiation

Of all the decision driving factors, the voice of the largest client of the POS system was considered the most dominant. As a top five wireless service provider in the United States, the client deployed the POS system in all of its physical stores across US time zones spanning from the East Coast to Hawaii. The general trend in the enterprise software marketplace pointed to a focus on lowering total cost of system ownership. After a few years of rapid expansion in the wireless telecommunication industry and continuous operating losses, the client began to feel increasing market pressures to achieve operating profits by the end of 2001. To further exacerbate the market pressures for profitability, the wireless telephone industry was becoming increasingly crowded with players of varying sizes, while the growth of new subscriptions began to show signs of entering a plateau. Amid the wireless telephone service trended toward being commoditized, the top client requested that GBTL provide a POS solution with an open architecture that would enable it to lower the total ownership cost by integrating the enterprise application more tightly into its IT infrastructure.

The request resulted in the client’s agreement to partially finance the cost of the product re-architecting efforts.

Internally within the company, both the product marketing and sales groups increasingly called for a new POS built with Java related technology. The product marketing group
needed a new architecture to reposition the POS system because the client/server platform was increasingly viewed as outdated in the enterprise software marketplace. The sales group also received similar market signals as they attempted to convince target customers in the crowded wireless industry to switch to the company’s client/server based POS system. On the product development side, the engineering group was in need of a new architecture that could provide better customization capability. The ongoing product upgrades that took place two to three times a year frequently encountered setbacks in cost and time as the existing customers demanded faster delivery time at lower cost. Unless it was to be re-architected with the so-called “latest” technology, the POS system appeared to enter the end of its lifecycle as a POS solution for the company.

2.4.2 Product Architecture – Second Generation POS System

The new product architectural platform is a distributed application with the processing of tasks distributed across multiple tiers of system components. As illustrated in Figure 3, the logical view of the new product architecture comprises three tiers of system components, namely the client-tier, middle-tier and database-tier. The client-tier consists of GUI (Graphical User Interface) forms that enable end-users of the system to interact with the POS system. The middle-tier consists of application server, LDAP server, and web server.* The database-tier is provisioned for data storage using Oracle’s Relational Database Management

* An application server is a server program in a computer within a distributed network that provides the business logic for an application program. The application server is frequently viewed as part of a three-tier application, consisting of a graphical user interface (GUI) server, an application (business logic) server, and a database and transaction server. (Source: www.lanyon.com/support/Glossary/Glossary-d.htm)

A LDAP (Lightweight Directory Access Protocol) server provides applications in looking services/devices and management users. It is essentially a “high speed” directory widely in use in the enterprise software products to maintain lists of people, i.e. login information and user privileges.

A web server is a computer program that serves the requested files which form web pages to the user's browser. (Source: www.freewebsiteproviders.com/glossary.htm)
System (RDBMS). The components in the client-tier and middle-tier will be constructed using Java programming language.

![Diagram: RMS 4.X High Level Three-Tier Architectural View (Logical View)]

From the programming standpoint, certain essential properties of Java as an object-oriented programming (OOP) language, such as encapsulation, inheritance, and polymorphism, make it arguably more superior than structural programming languages in general. The project team is convinced that the OOP language will provide the re-architected POS system with better customizability and maintainability than the original product. Given that the POS system undergoes a scheduled release in every six months, customizability and maintainability are among the most crucial product attributes.

From the system standpoint, the ability of the new product architecture to distribute task processing is purported to provide system scalability and robustness – this specific product
attribute is highly critical for enterprise systems with high transaction volumes and low levels of tolerance for system downtime. Note that the middle-tier of the system can be configured with multiple nodes of identical application servers, LDAP servers, and web servers. System scalability can be technically achieved by adding identical nodes and distributing transaction loads among multiple nodes. Accordingly, system robustness can be theoretically attained by provisioning load-balancing and fail-over mechanism among nodes for incoming tasks from the client-tier as well as by better managing and distributing the database connections between the middle-tier and the database-tier. Despite the promises of the new product architecture, it is important to recognize that the amount of system integration efforts required for the three-tier architecture will far exceed that for the existing client-server product architecture.

2.4.3 Project Planning

In the fourth quarter of 2001, GBTL officially kicked off the project to re-architect its POS system. The project was provisioned to deliver the next generation POS system using Java programming language and multi-tiered J2EE platform*. The project was scheduled for ten months from the project inception to general availability of the new system. The total cost of the project was budgeted for a low single-digit million dollars and would be partially funded by the key client who requested for a POS system with a more open architecture.

The general product development strategy was to maintain the functional requirements of the re-architected product. With the exception of the product to be built based upon new product

---

*J2EE (Java 2 Enterprise Edition) is an application server framework from Sun Microsystems for the development of distributed applications. The platform began to gain dominance over client/server system in the second half of the 1990s as Java, an object-oriented programming (OOP) language, and the J2EE platform became widely adopted by the software development community in the United States.
architecture, the visual “look and feel” of the re-architected POS system would remain identical to end-users. The next generation POS system would enable customers to reap expected benefits from a system with a more open architecture and yet would require no end-user training. Keeping the product functionality unchanged would translate into one-to-one porting of product functionality from old product architecture to new product architecture. The project team treated the constancy of functional requirement as a positive attribute because it was likely to preclude development uncertainties usually associated with unplanned product requirement changes.

<table>
<thead>
<tr>
<th>Time</th>
<th>10 months from project inception to the general release of final product.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>The lowest costs in the marketplace</td>
</tr>
<tr>
<td>Quality</td>
<td>No critical product defects with comparable product performance.</td>
</tr>
</tbody>
</table>

Table 2 - Key Project Attributes Defined by Internal Project Sponsor

In terms of systems and project management (Table 2), the internal project sponsor defined three project attributes, namely time, cost, and quality, which would drive the planning and implementation of the project under study. The company mandated the project team to pursue an aggressive schedule, starting the project from the project inception to general release of final product in ten months. The project team was also mandated to achieve the lowest costs possible in the marketplace. The project sponsor defined the quality attributes as the delivery of final product with no critical defects and comparable performance.

The project attributes prompted GBTL to explore ways to fulfill these first two attributes that were seemingly contradictory. The first project attribute, namely the time constraint, led to a
company’s decision to contract out a portion of the product development efforts. Mostly based upon internal technical evaluations of the new product architecture and partially on internal engineering preferences, the company decided to contract out the development of the client-tier of the system (Figure 3). Internal technical evaluations concluded that the client-tier is the least important strategically to maintain its engineering competency. Consisting of many GUI forms in the client-tier, the development of GUI components for the new POS system involved laborious efforts and was generally regarded as “unappealing” by software engineers. The cost constraint resulted in the decision to grant the external development contract to an offshore outsourcing vendor called VERT. Specifically for same constraint, the company also made a business decision to outsource the actual implementation of software testing with the internal software quality assurance group responsible for defining and verifying the quality of the implementation. The company’s decision to outsource certain product development tasks was a markedly significant event because it had virtually no experience with outsourcing activities.

The project under study intended to re-architect the product and port the major customer to new POS system in ten months. Figure 4 illustrates the planned product road map when the major customer was scheduled to switch from the first generation system (POS 3.5) currently in use by customers to the second generation system in (POS 4.0). Upon the switch to the next generation POS system, which would require an enterprise-wide deployment of the new system, the product would proceed with subsequent product upgrades in very six-month interval under the business mechanism very similar to that provisioned for the first generation product. Note that POS 3.5 was in a parallel development phase with a target release in the first quarter of 2002.
The project team utilized Microsoft Project for project planning and tracking. Table 3 contains the high-level master schedule for the project under study. Created through the joint efforts between the internal project manager and outsourcer’s project manager, the master schedule can be characterized as overlapping of sequential tasks. For instance, there is an overlapping of tasks between tasks 153 and 154. The master scheduled did not set any explicitly defined project milestones. The project was scheduled to begin on 12/10/01 and finish on 9/13/02. The schedule allocated 70.5 work days to the project inception and elaboration and 178 work days to the project construction. In essence, the project elaboration comprised product design efforts and the project construction product development efforts.
Table 3 – High-Level Master Project Plan*

In terms of human resource allocation, GBTL assigned the project as the primary work responsibility to fifteen staff members in the product development group. Among the assigned staff members, four were assigned to the entire project lifecycle with the remaining members working full-time on the project in specified periods ranging from three months to nine months. The allocation of human resource on the side of the offshore outsourcing vendor was unavailable to the project team because the vendor was engaged at predetermined fees.

---

*The ownership of tasks is grouped into two categories, where the tasks in green cells belong to the outsourcing vendor and yellow cells to the company of the project. The project plan was derived from the master schedule in under Section 6.1.
Note that the project schedule was considered highly ambitious by most project team members (Figure 5). The ambitious nature of the schedule can be indicated by a high degree of parallelism in the master project. The project timeline (Figure 6) exhibits a significant project overlap in the project construction phase between POS Beta-1 and POS Beta-2. In this case, POS Beta-2 (task 154) is scheduled to begin on 3/14/02 while POS Beta-2 Sign-off (task 153) is scheduled to complete on 5/31/02. This specific overlap illustrated the highly ambitious nature of the project scheduling. In other words, the project plan compressed two sequential phases of product release into parallel product development efforts.

* For access to the project timeline at a bigger scale, refer under Section 6.2
2.4.4 Project Implementation

The development manager of the first generation POS system was appointed the internal project manager of the cross-functional project team. The development manager was originally hired for administering the upgrades of a mature, established product. The job position required no general engineering background but only administration skill set. It quickly became clear to the project team that the internal PM lacked in-depth product knowledge and experience in managing new product development projects. Rather than focusing on proactively identifying and resolving project roadblocks, the internal PM turned weekly project meetings into unproductive sessions the team focused mostly on administrative tasks that could be easily accomplished via email communication.

The internal PM’s lack of required skill set for managing software re-architecting inadvertently resulted in the offshore outsourcing vendor, VERT, driving the project schedule and dominating the management of the project under study. There were multiple communication flows between the project team and the outsourcing vendor. Figure 7 displays the organization structure of the project team. The presence of multiple information flows was simply one of the indicators showing the organizational challenges facing the project under study. Amid the organization challenges and signs of imminent project slippage, several major personnel changes took place in February 2002 where a Program Manager was brought into the project team via external hiring and the company requested the PM of the outsourcing vendor be replaced. The internal PM was relegated to the general administration of project tasks before she was subsequently terminated by the company. Another major personnel change occurred around May 2002 where both System Architects
exited the project for unspecified political reasons and a new System Architect joined the project team via external hiring.

As the project elaboration scheduled to complete on March 20, 2002 lasted into late April, 2002, significant schedule slippage became increasingly imminent. A few important action items, including product deployment requirements and financial reporting, were discovered missing in the original master project schedule. The missing action items led to scope creeps that the project team thought to have been generally precluded by keeping the product functionality unchanged from the first generation product. These serious problems prompted the newly hired Program Manager to attempt stabilizing the project by conducting

---

* The Project Manager also functioned as the Manager of Software Development.
renegotiation with the outsourcing vendor and actively managing the project team*. The additional engineering time overhead required for merging parallel development efforts between POS-Beta 1 and POS-Beta 2 under the project construction caused further schedule slippage. The project planning tool failed to account for the additional time overhead as the original internal PM and outsourcing vendor’s PM attempted to meet the target project delivery date by aggressively increasing the level of parallelism among tasks without evaluating and managing emerging risks and uncertainties posed by such an attempt.

---

* Additional detail on the actions taken by the Program Manager can be obtained in the interview transcript in Section 7.1.2 – One-on-One Interview with Program Manager

---

Figure 8 – Actual Outcome of Product Road Map

By June 2002 the project team’s inability to meet the target project delivery date was essentially confirmed by various technical observations, ranging from mounting product defects to lackluster system performance to serious challenges in system integrations. As
shown in **Figure 8**, the schedule slippage resulted in cascading impacts on the product release plan. The reaction of the project under study to company’s need to meet customer’s demand for a product upgrade in every six to nine months was essentially that of playing a catch-up game by rolling the additional product features provided in POS 3.x releases into POS 4.x releases.

To put it in a different way, the Program Manager revised the product re-architecting project schedule into a three-phase project plan. The project team officially wrapped up the release of POS 4.0 in October 2002 and rolled incomplete project deliverables in the second phase of the project plan. A new set of project schedules were created to meet the target release date of POS 4.1. The project team aimed to port the key customer from POS 3.6 to the new system as POS 4.1. The project schedule for POS 4.1 is available in **Section 6.3 on Appendix A**. The project schedule for POS 4.1 was significantly more detailed than that of the master schedule (**Section 6.1 on Appendix A**) for POS 4.0. The greater level of detail in POS 4.1 schedule and the provision of clearly defined project milestones indicated the Program Manager’s efforts on actively managing the project. Unfortunately, the inability of the project team to overcome all existing technical obstacles on time as well as the emergence of additional technical problems resulted in another failure to meet the release date for POS 4.1. The failure to meet the release date for POS 4.1 resulted in a new set of project schedules for POS 4.2 with a new goal to port the key customer the new POS system from POS 3.7. Mainly due to schedule slippage aid the necessity to meet customer’ ongoing needs to add functionality to the POS system, the functional requirements that were previously regarded to a constant attribute essentially evolved to a moving target.
2.4.5 Project Outcome

The project under study lasted for almost twenty-one months, eleven months longer than originally scheduled, before development efforts were halted in August 2003. The project freeze was followed by numerous negotiations between GBTL and the key customer who initiated and agreed to finance a substantial portion of the POS system re-architecting project. Despite the fact that POS 4.2 was very close to project completion and almost ready for general release, the customer decided not to further pursue the software re-architecting efforts of the POS system.

The project was officially terminated in January 2004. GBTL subsequently transferred the product ownership of both the customized version of POS 3.x and POS 4.x source code to the key customer. Following this termination of the re-architecting project was the company’s decision to exit the POS system marketplace. The company subsequently sold another customized version of POS 3.x to another key customer. All ongoing business and activities related to the POS product has been confined to fulfilling technical support obligations to existing customers, ranging from small to medium sized, that are still using the POS 3.x system in their retail operations.

2.5 Chapter Summary

The project under study entails the re-architecting of an established enterprise software product. The project was plagued with schedule slippage and cost overrun and was eventually halted as the wireless industry entered a consolidation phase. Among the many project attributes, the project was particularly unique from the project organizational standpoint because it involved an offshore outsourcing vendor in developing a major
component of the POS system. As described in the Chapter Introduction, the main purpose of Chapter 2 lies its establishing a foundation for a case study. The case study will be conducted using SDM theories, methods, and tools. Simply put, the chapter establishes the context within which the analysis of the case, research on subject matters raised by the case, and potential prescriptions for the project and others of similar characteristics will take place.
Chapter 3: Survey Implementation and Findings

3.1 Chapter Introduction

The chapter is provisioned for the description of the survey implementation and the documentation of the survey findings. Starting with a brief overview of the case, it describes the survey implementation in terms of its survey targets, focus, and questionnaire. All responses to the survey questionnaire and the interview note with the program manager of the project under study are placed in Appendix B. Following the coverage on the survey implementation are the survey findings from the responses to survey questionnaire and the additional findings from survey follow-ups via email and telephone conversations with survey respondents. These survey findings form the basis for a detailed analysis of the project under study in the next chapter.

3.2 Case Overview

The case under study entails a re-architecting of an established enterprise software product, a point-of-sale (POS) system, at a mid-sized telecommunication solutions provider between 2001 and 2003. The original product was a client/server system and the re-architected product was based on three-tier J2EE platform. The product re-architecting project was driven by multiple stakeholders – a primary customer wanted a more open architecture for meeting its internal integration efforts on its backbone systems, and the company (GBTL) was in need of a competitive POS product in terms of better customization capability and better product positioning with more recent technology. GBTL’s view of the marketplace pressure led it to adopt a rapid product development lifecycle to meet target delivery dates.
and outsource the development of the client tier to an offshore firm (VERT) to lower development costs.

Originally scheduled to complete in ten months, the project encountered a series of setbacks, including significant schedule slippage and key personnel turnover. Key personnel turnover involved the replacement of original project managers from GBTL and VERT, and the withdrawal of original system architects in the middle of the project. After twenty-one months into the project, the key customer that drove the re-architecting initiative notified GBTL of its decision to not accept the re-architected POS system. This led GBTL to immediately halt and subsequently terminate the project under study.

3.3 Survey Implementation

The section describes the overall implementation of the survey for gathering information on problems encountered by the project as well as for identifying key project outcomes. The survey implementation entailed the identification of survey targets, the determination of survey focus, and the design of survey questionnaire.

3.3.1 Survey Targets

The main purpose of the survey was to collect cross-functional professional experiences and opinions on the project under study. The survey targets were chosen primarily based upon the organizational structure for the project team (Figure 7) and the high level of involvements by key members in the project. The survey targets consist of System Architects, Database Architect, Technical Analyst, Documentation Specialist, Database Administrator, Technical Consultant, Technical Leader and QA Manager. Table 4 delineates the role of each survey
target and the organizational unit to which he or she belonged, where all survey targets were internal members of the project under study and still remained with the company at the time of the survey implementation. Due to logistical reasons, the survey did not target former employees who left voluntarily or were terminated by the company. Among them was the original Project Manager who was terminated on the basis of poor job performance in March 2002, four months after the official kick-off of the project. The causes and reasons that led to the termination of the Project Manager are covered in great details in the next chapter. Her responsibilities for the project under study were transferred to a Program Manager, who came onboard via an external senior hiring in February 2002.

<table>
<thead>
<tr>
<th>Role</th>
<th>Department</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Architect (Original)</td>
<td>Enterprise Architecture</td>
</tr>
<tr>
<td>System Architect (Replacement)</td>
<td>Software Development</td>
</tr>
<tr>
<td>Technical Analyst</td>
<td>Software Development</td>
</tr>
<tr>
<td>Documentation Specialist</td>
<td>Product Documentation</td>
</tr>
<tr>
<td>Database Architect</td>
<td>Database Engineering</td>
</tr>
<tr>
<td>Database Administrator</td>
<td>Database Engineering</td>
</tr>
<tr>
<td>Technical Consultant</td>
<td>Performance Engineering</td>
</tr>
<tr>
<td>QA Manager</td>
<td>Software Quality Assurance</td>
</tr>
</tbody>
</table>

*Table 4 - Role of Survey Targets*

The offshore contractor, VERT, was not included in the survey because the paper intends to focus on not on outsourcing activities per se but challenges faced and lessons learned by the company, GBTL, in the re-architecting of an established enterprise software. Furthermore, the animosity between GBTL and VERT resulted directly from the project under study has rendered it impossible to get the offshore contractor to cooperate and participate objectively.

* The author functioned as Senior Performance Engineer for the project under study. His engineering responsibilities for the project included system integrations, configurations, and scalability on the re-architected POS system. He was a member of the Production Engineering – Performance Engineering group.
in the study. Given that the survey responses alluded to several major setbacks encountered by the project team, follow-up communications with some of the respondents were conducted via email and telephone.

### 3.3.2 Survey Focus

The general survey is intended for probing the history and nature of the project on the basis of system design and project management. By studying the history and nature of the project, the author strives for identifying key problems encountered throughout the product development lifecycle of the project. Furthermore, the author intends to research and evaluate if these problems could have been actively managed and/or positively influenced by the application of SDM (System Design and Management) frameworks, methods, and tools.

### 3.3.3 Survey Questionnaire

In order to ensure that the survey is free from the author’s inherent bias due to his heavy involvement in the project under study, the questions were specifically designed to be open-ended and not technically specific*. The questionnaire was provisioned for gathering diverse viewpoints across all engineering functional groups. Figure 9 entails the actual questionnaire distributed to and responded by its survey targets.

<table>
<thead>
<tr>
<th>General Instruction:</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am sending you a quick survey on POS Re-Architecting project. The survey intends to gather some</td>
</tr>
</tbody>
</table>

---

* The questionnaire was initially designed for a term project of ESD.36J System Project and Management. The survey questionnaire underwent multiple iterations focusing on probing the history and nature of the project on the basis of systems and project management. The initial survey was conducted in Fall 2004. Follow-ups on responses to the survey were conducted specifically for this chapter.
background information about the experience of each member involved in the project. I will likely to touch base with you as a follow-up.

I assure you that I intend to use the survey solely for the study of tools and methods for project management. I will filter out all business sensitive information accordingly.

**Survey Questions:**

1) Do you consider the project a success or a failure? Why?

2) What areas went wrong? What was your reaction if applicable?

3) What areas went right? What was your reaction if applicable?

4) What should the project team do differently if it could work on the same project again?

5) What’s your role in the project?

**Figure 9 - General Survey Questionnaire**

### 3.4 Survey Findings and Follow-Ups

This section is intended for capturing major problems and key outcomes of the project under study. The process entails the review of survey responses for common themes and the follow-up on selective issues raised by individual respondents in their survey responses. The next chapter covers further analyses of key survey findings and prescriptions of SDM methods and tools for addressing specific problems encountered by the project.

#### 3.4.1 Findings on Responses to Survey Questionnaire

The survey responses were carefully reviewed and probed for possible common themes or viewpoints. Based upon careful reviews of the responses, the respondents appeared to share many common views on the project under study. The common views of the survey findings
are tabulated in Figure 10. The detailed findings for each survey question are delineated in sections 3.4.1.1, 3.4.1.2, 3.4.1.3, and 3.4.1.4.

<table>
<thead>
<tr>
<th>Question 1: Do you consider the project a success or a failure? Why?</th>
</tr>
</thead>
<tbody>
<tr>
<td>The project was considered a failure by all respondents. The assessment was drawn from the fact that the re-architected POS system was never deployed in productions.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Question 2: What areas went wrong? What was your reaction if applicable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Four major things went wrong.</td>
</tr>
<tr>
<td>- First, the project schedule was considered to be too aggressive by all stakeholders, and the management team failed to adequately address the scheduling concern with specific management techniques, i.e. risk management.</td>
</tr>
<tr>
<td>- Second, the project failed to engage the primary customer in making key product design decisions, resulting in customer’s rejections of key designs after they were implemented, i.e. database schema redesign and thick client vs. thin client.</td>
</tr>
<tr>
<td>- Third, the product (client-tier) was overly architected due to inadequate product knowledge among internal System Architects, i.e. internal data exchange mechanism using Java InfoBus might not be necessary.</td>
</tr>
<tr>
<td>- Fourth, the project team encountered major organizational challenges working with an offshore outsourcing vendor, i.e. incompatible development process and ineffective information exchange and flow.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Question 3: What areas went right? What was your reaction if applicable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>No direct mention of what went right in the project. All cited valuable lessons learned at both the enterprise and functional unit levels.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Question 4: What should the project team do differently if it could work on the same project again?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two major weaknesses were identified for future improvements.</td>
</tr>
<tr>
<td>- First, the need for better communication and coordination among project stakeholders.</td>
</tr>
<tr>
<td>- Second, the need for product knowledge to make sound technical decisions.</td>
</tr>
</tbody>
</table>

Figure 10 - Tabulation of Survey Responses

3.4.1.1 Detailed Findings on Question 1 – Project a Success or Failure?

When asked if the project was a success or a failure, the respondents unanimously considered the project a failure because the re-architected POS system (POS 4.x) was never deployed in production. To put it simply, the end product was essentially rejected by the customer that sponsored the project. Almost all respondents attributed the significant schedule slippage and
cost overrun to the customer’s rejection on key designs. Almost two years after the project was initiated, the wireless industry as a whole had moved from a rapid expansion phase to a consolidation phase. By the time the re-architected product was ready for deployment, the customer was amid massive cost cutting efforts and had undergone several rounds of major layoffs.

In terms of product readiness for general release, a few respondents weighed in their diverse opinions based on their respective job responsibilities. The QA manager pointed out that POS 4.x never hit the fundamental quality threshold where the re-architected product was required to be “equal to or better than the legacy system built on PowerBuilder.” From the viewpoint of the Technical Analyst, who was responsible for translating product code and knowledge from the existing POS system to the new architecture, architectural design decisions were made without the customer’s adequate participation and consent leading to rejections of some key designs. The detail on key designs rejected by the customer is delineated in Section 3.4.2. These rejections resulted in additional schedule slippage and cost overrun as the project team had to allocate unanticipated time and resources for resolving design issues raised by the customer.

3.4.1.2 Detailed Findings on Question 2 – What Went Wrong?

According to the survey responses, it appeared as if many areas that could go wrong in the project under study actually went wrong. There was a general consensus that the project schedule was too aggressive and that the customer’s engagement in the re-architecting efforts was not sufficiently managed throughout the project lifecycle, particularly in the design stage of the project. The stakeholders’ concern over the aggressiveness of the project schedule was
not adequately addressed by the internal sponsor of the project. Equally detrimental, GBTL did not engage the primary customer adequately in making key design decisions resulting in costly rejections of key designs in the implementation phase.

The majority of the respondents attributed the project failure to the architectural design of the new POS system. Upon further investigation on the issue pertaining to the product being “overly architected or engineered”, all respondents referred it to flawed design decisions in the client-tier. Further detailed discussion on the design setback is provisioned in Section 3.4.2.2. Although many agreed that the product design was overly architected, they differed on the causes of the design problem. The author views the different opinions among the survey respondents on the causes of the design problem mainly as a reflection on their functional roles and amount of experience they had before the project started. The causes ranged from a lack of product knowledge among system architects and developers to the absence of attempts to verify viabilities of the architectural design prior to the development phase. A few respondents wished that there were a mechanism put in place to proactively detect and resolve technical problems arose from the overly architected design.

The most heated issue facing the survey respondents turned out to be the company’s decision to engage an offshore outsourcing contractor (VERT) in the product development. In addition to questioning the business rationale on outsourcing some of the development efforts, many complained about challenges working with the offshore contractor, ranging from methodology incompatibility and communication problems between the company and the contractor to the poor quality of work delivered by the contractor. Two respondents
pointed out that the company had virtually no experience in managing either the outsourcing of product development or the relationship with the offshore contractor.

3.4.1.3 Detailed Findings on Question 3 – What Went Right?

All survey responses to this question cited indirect benefits gained by GBTL and the project participants in the area of organizational learning. Organizational learning took place at both the enterprise the functional unit levels. At the enterprise level, GBTL gained valuable insights from managing outsourcing activities and believed that it would be able to better manage offshore contractors in the future. In terms of organizational learning at the functional unit level, many cited the acquisition of new technical knowledge and engineering practices associated with the design and implementation of three-tier J2EE platform. The engineering organization at GBTL could leverage the newly acquired knowledge and practices on products other than the POS system. Interestingly, a respondent who was privy to GBTL’s financial reporting pointed out a fact that the company met its short-term goal of generating recurring consulting revenue as the project was partially sponsored by one of its key customers. It is very crucial to note that there was no explicit mentioning of what had gone right in the project under study itself.

3.4.1.4 Detailed Findings on Question 4 – What Could Have Been Done Differently?

The survey responses essentially identified two major weaknesses that are deeply intertwined by the nature of the project. First, all respondents not only called for better communication between GBTL and its offshore contractors but also among the functional units within the company and between the company and the primary customer that sponsored the project. In
other words, the survey responses identified a real need to strengthen communication and coordination among stakeholders. Second, they pointed out the need for making design decisions in the context of the existing POS system and its customer base. In other words, software architects and offshore contractor needs to fully understand both the functionality of the existing product and how it was being implemented and utilized in productions. Given that some of project members possessed a good understanding of the existing product, poor technical design decisions solely due to the second weakness could have been prevented by better communication and exchange of ideas among all stakeholders in the project.

3.4.2 Findings on Design Setbacks

Follow-up communications with some respondents via email and telephone helped identify several major setbacks that contributed significantly to additional schedule slippage and cost overrun. The setbacks are captured based on their relations to the architectural design of the new POS system. The highlights of these major setbacks and their negative ramifications are outlined in Figure 11. The detailed descriptions of the setbacks are classified according to the architectural layout of the new POS system in the next section.

<table>
<thead>
<tr>
<th>Three-Tier Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Even though the architecture inherently required extensive system integration efforts, the original project plan did not explicitly allocate time and resources for such efforts. The project plan arrived at a 10-month delivery date by overlapping two phases of development.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setback:</th>
</tr>
</thead>
<tbody>
<tr>
<td>System integration was complicated by the failure of GBTL to actively manage the process and keep it in house. Excessive overlapping of “sequential” tasks further complicated the situation. The complication led to an official admission of significant, inevitable schedule slippage.</td>
</tr>
</tbody>
</table>
### Client-Tier
The original design called for (i) a thin client that would run on existing client hardware and incur lower deployment cost than the existing POS system, and (ii) retaining the existing user interface to minimize the need for end-user training.

**Setback:**
The client component evolved into a thick client design for meeting the second requirement without compensating for the first requirement. The primary customer objected to the design change because it required costly client hardware upgrade. The objection took place after the design was implemented resulting in costly attempts to reduce system resource required by the client component.

### Middle-Tier
Several key pieces of product functionality were not accounted in the design phase.

**Setback:**
The oversight was detected during system integration efforts, which essentially resulted in scope creeps even though the product functionality remained the same between old and new POS system.

### Database-Tier
The database component was rejected by the primary customer after it had been constructed.

**Setback:**
The primary customer was supposedly made aware of the engineering decision to redesign the database component for achieving greater scalability. Nonetheless, the design was rejected when the primary customer became aware of its ramifications during a system demonstration. The rejection forced the project team to discard the implementation and reverted to the existing database design.

---

**Figure 11 - Major Project Setbacks and Ramifications**

### 3.4.2.1 Three-Tier Architecture

As a direct consequence of bad oversight and poor planning on system integration efforts, the project encountered significant challenges and setbacks in integrating the three-tier architecture. First, in its attempts to stem ongoing project slippage, the project team made a major mistake on assigning the task of integrating client components to the offshore vendor. The decision was considered a major mistake because the value proposition of the offshore contractor lied mostly in its responsibility for delivering the client-tier components based on the product specification defined by GBTL. Second, the manipulation of the project plan to arrive at aggressive target dates resulted in significant overlaps of project construction phases...
proved to have far-reaching consequences on software development and system integration efforts. The creators of the project plan failed to evaluate and provision for the consequences of overlapping project construction phases. These overlaps proved to further complicate the integration efforts as they required manual, painstaking tasks to merge components from two overlapping construction phases*. 

3.4.2.2 Client-Tier:

The client-tier was severely impacted by two flawed design decisions. The flawed design decisions had such far-reaching consequences the POS system as a whole was extensively referred to as being "overly architected" or "overly engineered" by the project team. The first design flaw refers to the deviation of the client component from its original design intent as a thin client. Directly due to the first design flaw to opt for a thick client design, the second design flaw took place where the unproven Java InfoBus mechanism for internal data exchange mechanism was chosen for the design.† Both design flaws gave rise to a thick client design that required client hardware with considerably greater system specifications, mainly faster processors and more physical memory.‡ The primary customer of the POS system vehemently rejected the design because it did not anticipate costly hardware upgrade. The project team was forced to explore ways to lower system resource requirement within the constraints posed by the thick client design.

* The overlapping of construction phases is discussed in Chapter 4.
† The InfoBus is a standard extension API consisting of the javax.infobus package. It is an API that defines Java classes and interface for data exchange operations. The InfoBus is provisioned for enabling data exchange between Java objects using Producer/Consumer model.
‡ The thick client design was commonly referred as Java Swing client because it consisted of building graphical user interfaces using Swing classes.
3.4.2.3 Middle-Tier:

Despite the fact that product functionality remained constant, scoop creep managed to take place as the original design failed to identify all key functionality elements required to deliver the original product functionality. For instance, the design and implementation of user authentication for access privileges was incorrectly treated as a "clean slate" requirement based on an incorrect assumption that the system would be deployed as a new installation. The oversight resulted in missing functionality for migrating existing users to the new POS system during the switch over in productions. Note that there were thousands of application users in the existing POS system used by the primary customers. The other example was the poorly thought out design in the deployment of client components to physical store locations during the initial switch over and in subsequent upgrades.

3.4.2.4 Database-Tier:

One of the major goals of the re-architecting of the POS system is to provide greater system scalability for meeting the business expansion anticipated by the primary customer. The system load was anticipated to grow by 60 percent in a one-year period. This specific requirement led to an engineering decision to completely redesign the database component, which involved major changes in the database schema and ways to retrieve and update data. In the design phase, the database engineering group strongly pointed out the significant ramifications of the database redesign as the design decision would require the conversion of an extensive amount of production data into the new database design. The project team proceeded to implement the database redesign under the notion that the design and its ramifications were properly conveyed to and approved by the primary customer. The project
team was caught in a total surprise when the database design was adamantly rejected when a beta release of the new POS system was unveiled at a client summit with the primary customer. The rejection left the project team with no choice but to revert back to the original database design. The ramification of the rejection was extremely significant because it severely impacted the project team's ability to meet a major project goal, which was to deliver a POS system with greater system scalability.

3.5 Chapter Summary

The implementation of surveys yielded great insights on the challenges and set-backs encountered by the project under study. The many things that went wrong and were of great concern to the survey respondents can be generally grouped into two categories, namely the management area and the engineering area. The management area pertains to leadership, business decision-making, and design of organizational structure and processes. Some of the management issues raised by the survey respondents are universal in nature because they are related to company’s setting realistic, attainable project schedules and managing business challenges associated with offshore outsourcing activities. The engineering area comprises product design and development (PDD) and systems and project management (SPM). PDD involves the creation and selection of design concepts and making sound technical decisions. SPM deals not only with managing product/system development process but also with identifying and managing uncertainties posed by engineering projects. In short, the survey findings form the basis for analyzing and prescribing SDM framework, methods and tools to address and resolve specific problems facing the project under study in the next chapter.
4 Chapter 4: Integrative Implementation Framework

4.1 Chapter Introduction

This chapter intends to create an integrative analytical approach in a form of an implementation framework for analyzing the project under study. The implementation framework is integrative because it comprises both general management areas and engineering areas. Offshore outsourcing of application development is recognized as an important factor in establishing context specificity for the implementation framework. Taking the specific context into account, the chapter creates an implementation framework specifically for software product re-architecting projects engaging offshore outsourcing vendors in the application development efforts. The implementation framework is applied to the analysis of key issues of the project under study tabulated from the survey findings in Chapter 3. The overall approach is highly constructive because it provides an integrative way to analyze the project study and establishes recommendations that are applicable to the re-architecting of enterprise software products within the aforementioned context specificity.

4.2 Context Specificity

The chapter emphasizes the importance of context specificity for a detailed analysis of the project under study. The stance of “one size does not fit all,” the paper rejects the notion of a project is a project is a project syndrome. The stance is applicable to the proposed implementation framework for the project. Although all real-world projects have a goal, a budget, and a timeframe, they tend to differ in various ways. For instance, the differences can lie in project size, product type, maturity of technology involved, and geography of project
team. Blindly applying a specific project management style and technique without understanding what makes a project distinctive in its own way often leads to project failures.

The paper views the creation of context specificity as a very useful approach for setting the analytical boundary and deriving an analytical framework for the project under study. Establishing context specificity for a case provides an effective way to understand the uniqueness of a project. When established in a valid way, context specificity enables the analysis of a project in a systematic way and the application of SDM methods and tools for addressing and resolving issues faced by the project.

The case story and survey findings in the previous chapters indicate a few key factors that define the uniqueness of the project under study. The key factor which the paper considers the most prominent is the engagement of an offshore outsourcing vendor in application development efforts. The company was notably inexperienced in such an undertaking. As delineated by the survey findings, many of the challenges and set-backs encountered by the project under study can be associated with GBTL’s interfacing and collaborating with VERT, its offshore outsourcing vendor. The subject of offshore outsourcing in the area of application development deserves further discussion in Sections 4.2.1 and 4.2.2. Another key factor is the implementation of new technology in the re-architecting project. The company started the project without any in-depth knowledge of or real hands-on experience with the new technology. Section 2.4.2 provides a detailed description of the specific technology selected for the project. The project’s initial failure to assess the risks/uncertainties posed by this aspect and its failure to provision steps to mitigate them resulted in numerous implementation setbacks.
4.2.1 Offshore Outsourcing – Overview

Offshore outsourcing of professional services is a highly contentious issue in the US. There are essentially two opposing views on the subject matter. Those with a favorable view believe that the phenomenon makes US goods and services more competitive in the global economy, whereas those with unfavorable view contend that it is purely a labor arbitrage that results in permanent job losses and associated export of intellectual property.* In the paper, the author intends to keep his viewpoint on this subject matter private and expeditiously adopt a neutral stance for the analysis of the project under study.

Offshore outsourcing embraces the concept of a “24-Knowledge Factory”, which involves “a globally distributed work environment in which members of the global team work on a project around the clock; each member of the team works the normal workday hours that pertain to his or her time zone. At the end of such a workday, a fellow team member located in a different time zone continues the same task.”2 Among the simplest forms of such organization are 24-hour call centers and customer support centers. The outcomes of this simplest form of offshore outsourcing deals have been mostly positive because they essentially involve effective execution of steps and operational requirements clearly defined by outsourcing customers. The work environment is operated under a set of well-documented standard operating procedures and equipped with information repository of log activities for subsequent retrievals and updates by personnel at centers in different time zone. This type of

* IP refers to IP comprehensive technological, operational, and business methods that enable workers to be effective. It is pointed out by an individual at Stanford and MITRE Corporation that export associated with offshore outsourcing occurs often “without any corresponding generation of income in the U.S., and without any corporate tax being paid on what are effectively export sales.”
outsourcing is enabled by personnel fungibility, where employees are generally interchangeable, and problem solving involves mostly a low level of required coordination between employees on any customer’s problem or work process in general. In essence, the first wave of offshore outsourcing in IT areas with personnel fungibility and low level of required coordination among employees.

The next wave of IT offshore outsourcing centers heavily around application development – an area where skilled employees are not interchangeable and product domain knowledge is crucial for ensuring effective product design and development. The offshore outsourcing of application development can be divided into two categories. The first category encompasses outsourcing of in-house application where companies decide between developing IT solutions in-house and buying them from software vendors for enabling their business operations. The second category involves the outsourcing of the development of, and in many cases also the design of, software products for commercial sales to third parties by software vendors. In this category, outsourcing activities range from actual coding of specific software components to both software design and development efforts. The second category essentially pertains to offshore outsourcing of product innovation – the area has been considered by most as not “offshorable.” Hence, companies need to understand the potentially far-reaching ramifications of their pursuit of offshore outsourcing on their core competence as creators of software products.

4.2.2 Offshore Outsourcing – Trends and Statistics

Amid increasing pressures on cost reductions, businesses in the developed world increasingly turns to offshore outsourcing vendors as a feasible source to lower costs. Successes in the
offshore outsourcing of call centers and customer support centers fueled the confident levels of business decision makers and led them to explore the feasibility of outsourcing other more complex business functions. The business functions targeted include research and development and software application development - functional areas that have been traditionally viewed as high value-added, critical to the organizational cores.

Figure 12 - Offshore Growth: Gartner's Projections for Offshore Outsourcing

According to Gartner’s report on offshore outsourcing growth, formally unveiled at its 2005 Outsourcing Conference in Los Angeles, there are less than 5% IT jobs in the US and other developed countries which are currently being outsourced to offshore outsourcing vendor. The report anticipates a “tectonic shift” in the offshore outsourcing landscape where the number will rise to 30% by 2015. Figure 12 shows that offshore outsourcing expenses in the areas of R&D and application are projected to exceed $10 billion and $50 billion in 2004 and 2010 respectively.
Interestingly, the enthusiasm among US businesses to outsource functional areas beyond relatively basic, low-risk business operations is not entirely free of worries. According to *InformationWeek*’s survey of 300 business technology executives in Figure 13, their outsourcing worries in general were in direct contrast with the seemingly promising future of offshore outsourcing as its trends toward high value-added and mission-critical areas. Note that the survey covers both onshore and offshore outsourcing. The top three outsourcing worries are (i) the quality of work, (ii) high or unexpected costs, and (iii) security of data or physical assets. These top three worries imply a lack of confidence among business technology executives in the ability of offshore outsourcing vendors to deliver on their intended outsourcing deals. Note that these worries are confined to project worries and not

---

* Data source: InformationWeek Research Priorities 4Q 2003 study of 300 Business-Technology Executives
strategic worries, such as loss of internal capability or market initiative. The survey result is particularly revealing because it implies a lack of general awareness among business technology executives of the potential adverse impacts of offshore outsourcing on their companies' internal capability.

The paper views the gap between these outsourcing trends and worries as a revealing testimony on the underlying challenges facing offshore outsourcing activities beyond its current, simplest forms of business operations. Furthermore, the paper believes that some of these challenges are exhibited in the project under study. The aforementioned gap implies that effective executions of offshore outsourcing of business activities further up the value chain take on a new complexity. If the new complexity is not managed properly, these types of outsourcing undertakings can easily result in poor quality of work and skyrocketing costs, and impact the ability of outsourcing customers to secure their data and physical assets.

4.3 Implementation Framework

The paper proposes an implementation framework for interpreting key events and diagnosing major setbacks encountered by the project study. Note that the proposed implementation is heavily driven by the context specificity of the project. As indicated by the survey findings in Chapter 3, the project calls for an integrative approach to address the major setbacks and key events it encountered. These events and setbacks can be categorized under general management and engineering, where the latter decomposes into product design and development (PDD), and systems and project management (SPM). In addition, the framework intends to illustrate how these two specific areas, namely general management and engineering, are closely intertwined. Using the proposed framework, the paper seeks to
address these key events and major setbacks by applying SDM frameworks, methods, and tools.

Figure 14 - Implementation Framework for Enterprise Software Re-architecting

As illustrated in Figure 14, the analytical framework to be used for the analysis of the project under study encompasses general management and engineering. Note that the framework is driven by context specificity – the offshore outsourcing of application development. Under the general management, the framework emphasizes the importance of corporate strategy and business decision making, and organizational processes and structure. The placement of engineering directly above general management in a pyramid structure indicates an integrative approach, where the robustness of general management is the prerequisite to the efficacy of engineering. Under the engineering layer, product design and development (PDD) and system project management (SPM) are in constant interactions. SPM entails the planning and scheduling of project tasks and PDD the actual execution of design and development work.
4.3.1 Corporate Strategy and Business Decision Making

The paper asserts that a carefully thought-out outsourcing strategy by a company and the ability of a company to evaluate the feasibility of a specific software application for offshore outsourcing are the two most important areas in the planning stage of any given project. These two areas enable a company to anticipate challenges facing an outsourcing project and devise its outsourcing management approach accordingly. For a company that is completely foreign to managing offshore outsourcing of application development, the process is more likely to prompt a company to investigate potential impacts of impending outsourcing activities on its existing organizational processes and vice versa. Among the outcomes of the process may include making changes to its organization structure and processes and aligning existing PDD and SPM processes with outsourcing management approach.

The paper advocates the importance of a carefully thought-out offshore outsourcing strategy, and it argues against a company’s making its decisions on offshore outsourcing of software application development solely or heavily on the aspect of cost-savings. Offshore outsourcing vendors typically tout their ability to achieve significantly lower costs, especially in the area of programming tasks. The ability of these vendors to offer lower bids arises from their leveraging labor arbitrage from a vast supply of skilled, educated labor at lower wage levels. The most common mistake on cost calculations made by a company in its opting for the offshore-outsourcing of application development is that it incorrectly assumes that “the savings will match the salary differential” between offshore and in-house wages. Cost savings in software programming generally constitutes less than half of the total costs. The remainder costs are incurred in the areas of analysis and design, testing, system integrations,
and other tasks required in software product development efforts and project management efforts.

In addition to devising a carefully thought-out offshore outsourcing strategy, a company needs to possess a solid capability in systematically assessing the feasibility of an application development project for offshore outsourcing. It must be made aware that not all types of application development projects are suitable for this specific type of delivery model. The company needs to understand the inherent challenges facing offshore outsourcing and acquire the capability to methodically access the feasibility of their application development projects for outsourcing. The biggest mistake that can be made by decision makers with no outsourcing experience or little software development experience is to expect offshore vendor providers to behave like a temporary agency or a seller in spot markets that supplies a company with additional human resource capacity or input commodity. In short, a company needs to develop an internal competency in evaluating the feasibility of its application development projects for offshore outsourcing.

4.3.1.1 Offshore Outsourcing Strategy

A company should always do complete due diligence on its outsourcing strategy for it to realize potential gains of offshore outsourcing of application development. In devising its outsourcing strategy, the company needs to focus on its skills and knowledge sets, and not on products or function. In other words, the company should focus on its core competencies by looking at its "intellectual skills or management systems that actually create a maintainable competitive edge," and build dominating skills beyond that of product-oriented skills that customers will value over time. Based upon its core competencies, the company can
establish a set of outsourcing expectations and creates criteria for identifying and selecting outsourcing vendors. The process of establishing its own set of outsourcing expectations forces the company to identify specific goals it intends to achieve from outsourcing undertakings. The identification of specific goals in turn drives the criteria for identifying candidates and selecting the most suitable vendor.

Outsourcing of product development generally results in a company’s dependency on its offshore outsourcing vendor, ranging from dependency on knowledge to capacity. Figure 15 describes outsourcing in terms of two dependency classes and the corresponding skills requirements. The two dependency classes are dependency for knowledge and dependency

Figure 15 - The Skills Required to be Dependent for Capacity Compared to Those Needed if One is Dependent for Knowledge

* The skills requirements and dependency types are originally provisioned as a make-buy decision process for the automobile manufacturing industry (Fine & Whitney, 1996)
for capacity. In the former, a company lacks required skills to develop a product in-house and therefore chooses to rely on its outsourcing vendors. In latter, a company possesses required skills but decides to outsource for capacity reasons. If a company is to be dependent upon its outsourcing vendor for knowledge, it needs to assess the level of importance of such knowledge to its core competence and whether such knowledge can be gained if it decides to bring the product development back in-house. In terms of dependency for capacity, a company needs to understand the minimal level of cost effectiveness for outsourcing on the basis of capacity. Note that the level of knowledge dependency decreases and required skills for successful outsourcing increases as they move from dependency for knowledge to dependency for capacity.

Figure 16 - Matrix of Dependency and Outsourcing
In addition to devising an outsourcing strategy in terms of dependency classes and required skills, a company should seek to understand the risks of outsourcing. Figure 16 exhibits four types of outsourcing risks in a 2x2 matrix of dependency and decomposability of an outsourcing item, where the description of the outsourcing risks is as follows:

- Quadrant A: Suppliers [outsourcing vendors] could supplant a company because they have as much or more knowledge.
- Quadrant B: A company understands it [product component considered for outsourcing] and can integrate it into its process or product. Suppliers can be obtained from several sources. Buying [Outsourcing] enables a company to divert attention to areas where it has competitive advantage.
- Quadrant C: A company does not understand what it is buying [outsourcing] or how to integrate it [product component considered for outsourcing].
- Quadrant D: You know how to integrate the item so you may retain competitive advantage even if others have access to the same item.

It is important to note the decomposability of an item is closely related to product design and development. A product is decomposable if it can be made into a simple form where each component can be easily described as a functional unit independent from the original system and has a set of clearly defined interfaces.

In short, the paper proposes offshore outsourcing strategy as a two-step process – first, a company needs to determine the dependency class by understanding the reasons behind its decision to pursue offshore outsourcing; second, a company needs to develop the capability to identify which specific quadrant of outsourcing risks its potential outsourcing project is
likely to fall in. The proposed two-step process for outsourcing strategy is very crucial because it can enable a company to provision for required skills and potential risks.

4.3.1.2 Feasibility of Application Development Projects for Offshore Outsourcing

The feasibility of an application development project for offshore outsourcing is driven by key characteristics pertaining to its requirements for product design and requirement (PDD) and systems and project management (SPM). For instance, software application to be developed that involves well-proven technology and well-defined business requirements is more feasible for offshore outsourcing than that with fluid business requirement and unproven technology. The same argument applies to an organization with well-defined organization structure and processes and one without such characteristics. In other words, companies must understand which key characteristics to be used for determine the feasibility and comprehend the true conditions of these key characteristics.

By stipulating the importance of outsourcing feasibility, the paper recommends a diagnostic framework for evaluation the feasibility of application development for offshore outsourcing. Figure 17 is the diagnostic framework is proposed by NASSCOM, the prominent trade organization for IT software and services firms in India, and McKinsey, the world leading management consulting firm, and is adopted by WIPRO, one of the major IT offshore outsourcing vendors headquartered in India. The NASSCOM-McKinsey framework enables a company to assess the “outsource-ability” of an application development based on two dimensions - management requirements and interaction requirements. The dimension of management requirements are under the direct influence of the stability of the requirements themselves and the number of interfaces to external systems. The dimension of interaction
requirements is driven by the amount of input from customers and other third parties and the level of internal interactions. The framework entails five diagnostic outcomes, ranging from “not offshorable” to “highly suitable”. While there are ongoing debates on what constitutes “not offshorable”, there is a general agreement that application development for mainframes falls into the “highly suitable” category.

![Diagram](image)

**Figure 17 - Diagnostic Framework: Offshore Outsourcing of Application Development**

*The framework for offshore outsourcing of application development by NASSCOM McKinsey Study is found in *Outsourcing 101, WIPRO White Paper* [www.wipro.com/whitepapers]. NASSCOM is India's National Association of Software and Service Companies, the premier trade body and the chamber of commerce of the IT software and services industry in India. NASSCOM's 900 plus member companies are in the business of software development, software services, and IT-enabled/Business Process Outsourcing services and comprise over 90% of the revenues of the software industry in India. McKinsey & Company is the world premier management consulting firm.*
4.3.2 Organization Structure & Processes

The paper asserts that organizational design, in terms of the organization structure and processes, and information infrastructure, required by application development are very different from those for the outsourcing of call centers and network operation centers. In order for software development efforts to take place effectively in the setting of a "24-hour Knowledge Factory", one needs to comprehend and overcome constraints and challenges inherent in an existing organization design.* Put differently, contemporary organization design was created without the context of globally dispersed workforce and offshore outsourcing. New paradigms is likely to emerge in the long run as software development efforts attempt to realize gains from shortening the absolute amount of time via a globally dispersed project team working around the clock. A new paradigm must facilitate knowledge management where tasks can be shared among engineers in different geographical locations and time zones with any significant overhead incurred in explaining their work to their counterpart in the next shift.⁹

The ability to design an organization structure with processes that can enable effective communication in a globally dispersed and temporally separated project team is extremely crucial for offshore outsourcing of application development. Organization structure and processes heavily dictates and influences the effectiveness of communication in settings involved offshore outsourcing activities. According to a survey of 50 executives, the

---

* Organizational design shapes the structure of an organization. It is determined by strategic grouping of people to achieve intended focuses, and its drives the organizational behavior and processes. For instance, an organization design with grouping by activity put people with similar job functions together; in this case, the organization behavior is characterized by activities concerned with specific job functions. The most commonly deployed organization structures are (i) functional organization structure (the oldest form), (ii) business or product division structure, (iii) geographic organization structure, and (iv) matrix organization structure (i.e. business/functional matrix) (Source: Ancona et al, Module 2.)
effectiveness of off-shoring activities depends heavily upon the ability of everyone involved to communicate with each other.\textsuperscript{10} The ability to communicate involves two areas – the facilitation of the actual acts of information exchange by the use of collaboration technologies and the management of knowledge resulted from information exchange between the outsourcing customer and outsourcing vendor.

The paper agrees with the call for a communication strategy for offshore development project.\textsuperscript{11} The management of knowledge is more challenging than the facilitation of the actual acts of information exchange. Knowledge management involves the transfer of knowledge within a project team via formal and informal channels and the supervision of the act itself. The former is considerably more challenging as it requires making changes to organization structure processes. Organization structure and processes need to be aligned to facilitate effective information exchange and knowledge management using knowledge management tools. The tools include document repository system for timely information update and sharing, and shared application for project and bug tracking. The latter can readily take place by deploying collaboration technologies and infrastructure links. Among the most commonly used communication techniques are conference calls, email, chat tools, and video conferencing.

Given that software application development is a collaboration-intensive process, the offshore dimension further increases the level of collaboration intensiveness. This leads the paper to embrace the assertion that, in order for offshore outsourcing to be effective and successful, “a revised onshore process is a prerequisite to a successful offshore process.”\textsuperscript{12} It is wise for a company to identify, if not speculate on, the potential impacts of offshore
outsourcing activities on its existing PDD and SPM processes, and make changes to its onshore process accordingly. For instance, a company will not do well on offshore outsourcing of its application development if it excels on creating and managing software products via ad hoc methods with heavy uses of informal channels for information exchange and informal knowledge management. It is always wise that a company, in its pursuit of this type of outsourcing, revises its “onshore process” while it still develops its software applications in-house to meet the basic requirements of extensive collaborations via formal channels.

4.3.3 Systems and Project Management

This section intends to show that SPM takes on a new level of complexity with offshore outsourcing activities. It is absolutely important to note that software development tasks other than those of programming require extensive coordination of information flow on product requirements between its offshore outsourcing vendor and customers by the company. The challenges and setbacks encountered by the project under study, as delineated in the previous chapters, clearly show that offshore outsourcing of software application development actually present an additional layer of challenges to the SPM aspect. In addition to having to deal with SPM challenges typically anticipated in most onshore application development projects, the offshore outsourcing dimension exacerbates the intensity of these so-called “typical” SPM challenges, as well as creating additional challenges that are unique to the offshore outsourcing context. Unless these “typical” and additional SPM challenges are properly managed, offshore-outsourcing of software product development is more likely to encounter poor product quality, cost overrun, and schedule slippage.
The majority of engineering projects can be generally described by any one of three SPM strategies – instructionism, learning, and selectionism. These fundamental SPM strategies are defined according to their assumption on information adequacy (Table 5) and key methods recommended. The recommended key methods consist of SPM and PDD approaches. The SPM approaches are sequential (a.k.a. waterfall) approach and iterative (a.k.a. spiral) approach. The PDD approaches include rapid prototyping and set-based approach*. Instructionism assumes adequate information and applies critical path planning, which is applicable only under the sequential approach, and risk management as key methods. Instructionism strategy is highly suitable for periodical upgrades of enterprise software products where the information of product design and its implementation capability is generally adequate. The remaining two SPM strategies, learning and selectionism assume inadequate information but differ in their selection of key methods. Both strategies are used in projects that involve new product development and/or face inadequate information in areas such constantly evolving product requirements and the feasibility of new technology. Given

* Set-based approach is generally a very costly PDD approach to implement and hence seldom used in software application projects. The PDD approach is prevalent at Toyota, where its designers think about sets of design alternatives vs. pursuing one alternative iteratively until they come to a final resolution. “Delaying decisions, communicating ‘ambiguously,’ and pursue excessive numbers of prototypes, enables Toyota to design better and faster” than the common practice of iterating. (Source: Ward et al)
that “one size does not fit” when it comes to SPM approaches, it is absolutely crucial that a project’s level of information adequacy is well understood and a SPM strategy with appropriate methods and tools are correctly applied.

In addition, the paper proposes the use of time-based control, such as deadlines, milestones, and other time controls. Note that traditional productivity measures, which imply rates of production, for non-software product are not applicable to software products. Attempts to measure software progressive via a productivity measure, such as the use of lines of code (LOC), ignore the inherent uncertainties posed by the nature of information adequacy. Contrary to establishing delivery time as merely targets for a project team to strive for, regardless of their opinion on the achievability of these target dates, time-based control uses time to measure project progress and encourages team work across functional groups. As it has been observed in various complex development projects, the use of time-based controls can used to pace a given project in relation to its overall time limits and encourage cross functional collaborations. If a project manager can obtain commitments on deadlines from a project team, team members are more likely to work toward meeting their targeted deadlines. The use of milestones is effective for tracking the progress of key deliverables and creating urgency to remove any roadblock encountered along the way.

The paper advocates the use of SPM techniques to manage challenges posed by inadequate information. Information inadequacy gives rise to uncertainties in specific project elements and therefore creates unknowns for a project. There are two types of uncertainty – know unknowns and unknown unknowns (“unk-unks”). SPM can apply specific PDD methods to manage both types of uncertainty though it is usually much harder to manage unknown
unknowns. Risk management technique under the learning strategy pertains to the management of known unknowns. The management of uncertainty, particularly in the timely identification of both types of unknowns, can increase the likelihood of a project to succeed. Hence, the management of uncertainties will receive additional attention in the coverage of PDD in Section 4.3.4.

In the context of offshore outsourcing of application development, the most common pitfall is the abdication of project management by customers to their offshore outsourcing vendors.\textsuperscript{17} As noted in Figure 17, only those application development projects with low management requirements and interaction requirements can apply the instructionism approach as its SPM strategy. For the remainder of application development project types, SPM needs to assume inadequate information and actively manage uncertainties that are inherent in these project types. Given that the selectionism approach is very costly to apply and most cost reduction is considered an important factor in offshore outsourcing, the paper argues that the learning approach is the most relevant SPM strategy within the given context specificity.

4.3.4 Product Design and Development (Software Engineering)

The paper argues that the key to effective design and development of software product lies in managing the inherent uncertainties posed by a project. The inherent uncertainties of design and development of software products are directly due to inherent difficulties faced by the software engineering field.\textsuperscript{*} Effective management of software product design and

\textsuperscript{*}According to Frederick P. Brooks, arguably the most influential figure in the software engineering field, the software technology faces two types of difficulties, namely the difficulties inherent in the nature of software and the difficulties in the production of software. The former is called the "essence" and the latter the "accidents."
development (PDD) requires an adequate understanding of specific risks posed by these inherent uncertainties. Software PDD risks fall into two general categories – requirement risks and implementation risks. Hence, it is important to manage PDD risks that have potentially significant impacts on the success of software product design and development.

Figure 18 illustrates a decomposition of risk management. The first level decomposition leads to four components. Risk identification requires an awareness of uncertainties from which specific risks emerge. Risk assessment involves an adequate understanding of risks identified and their potential impacts on project success. Risk mitigation depends on accurate

The essence is a construct of interlocking concepts, ranging from data sets, relationship of data, algorithms, and invocation of functions. Brooks opined that the difficulties lie in the “essence” and not the “accidents,” where the design and testing of the conceptual construct are much more complex than the actual coding and testing of the representation of the specification. He pointed that past breakthroughs had provided solutions mostly for the accidents. The author hoped that future breakthroughs would provide solutions to the essence. Past progress has reduced the risks faced by the accidents whereas future progress depends upon the addressing of the risks faced by the essence of software development. As of today, eighteen years after Brook’s “No Silver Bullet: Essence and Accidents of Software Engineering” has been published, the real difficulties of software development still reside in the “essence” as defined by Brooks. In short, software developments are still plagued with same issues related the design, specification, and testing of the specification.

* Source: Lecture 9 (Slide 20) of ESD.36J Systems and Project Management by Dr. James Lyneis. According to Dr. Lyneis, Professor Warmkessel is the primary source of the decomposition.
risk assessment and application of specific techniques to prevent specific risks from realization or lessen their impact if prevention is not entirely possible. Risk tracking and handling pertains to ensuring systematic follow-up and reverting to risk mitigation if specific risks remerge.

Requirements risks pertain to challenges in gathering of accurate requirements and managing requirement changes throughout a product development lifecycle. Among the requirements risks include (i) customers think they know what they want but actually don’t know what they need, (ii) incorrect assumptions of customer’s needs frequently made by developers, (iii) inconsistent expectations between customers and stakeholders, (iv) customer’s lack of interests in defining their requirements, and (v) risks in the requirements due to the implementation of rapid development approaches.18

A feasible way to mitigate requirements risks is to implement requirements engineering. Proposed under the premise that a software development project with incorrect requirement will fail, despite a flawless execution in coding and testing, requirements engineering is provisioned to steer a software development project toward producing the right software.19 Requirements engineering considers quality requirements to be more important functional requirements. Functional requirements tend to receive much greater attention than quality requirements in software design and development. The attribute of quality requirements are system level requirements such as reliability, performance, scalability, robustness, security, and ease of use. The failure to focus on quality requirements often leaves a software project team with no choice but to re-architect its product under development. Requirements engineering also points out the reality of design ambiguities in software projects and

77
recommends they be treated as uncertainties with plans for tracking and handling them throughout the project life cycle as remedy.

Taking the viewpoint that software design and development involves the application of knowledge owned and the acquisition of knowledge not yet own, implementations risks emerge from uncertainties of a project team’s ability to apply and acquire knowledge required for meeting product requirements. Implementation risks also emerge from the need for partitioning PDD efforts into discrete tasks for multiple team members to work on and putting together programming outputs of the aforementioned discrete tasks into a cohesive, functional software product.

Assuming that requirements risks are adequately mitigated, implementations risks can be mitigated by provisioning a timely, cost-efficient feedback mechanism for verifying product design effectiveness. The paper proposes design structure matrix (DSM) and “enlightened experimentation” as techniques for managing implementations risks. DSM is applied as information exchange model to analyze development tasks, where interdependent and coupled tasks are identified and closely managed. “Enlightened experimentation” involves designing and conducting experimentation as early as possible to enable early failures and generate knowledge from lessons learned. Rapid prototyping and frontloaded development are two forms of “enlightened experimentation” techniques. Both techniques can be concurrently applied because they complement each other; DSM is about information flow structure and “enlightened experimentation” is about information content generation.
<table>
<thead>
<tr>
<th>Requirements Risks</th>
<th>Challenges:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Gathering of accurate product requirements</td>
</tr>
<tr>
<td></td>
<td>• Managing ongoing changes in product requirements throughout product development lifecycle</td>
</tr>
<tr>
<td>Proposed Technique:</td>
<td>• Requirements Engineering</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Implementations Risks</th>
<th>Challenges:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Uncertainties of project team’s ability in applying and acquiring knowledge required for meeting product requirements</td>
</tr>
<tr>
<td></td>
<td>• <em>Note: Software development is viewed as the application of knowledge owned and acquisition of knowledge not yet owned.</em></td>
</tr>
<tr>
<td>Proposed Techniques:</td>
<td>• Design Structure Matrix</td>
</tr>
<tr>
<td></td>
<td>• “Enlightened Experimentation”</td>
</tr>
</tbody>
</table>

**Table 6 - Summary of PDD (Software Engineering) Risks**

Taking into account offshore outsourcing as the most prominent component of the context specificity, a company that chooses to outsource its software product design and development efforts to offshore must actively control both requirements risks and implementation risks (*Table 6*). The paper asserts that the magnitude of both types of risks is greatly intensified by the underlying characteristics of offshore outsourcing – namely a globally dispersed and temporally separated project team. Simply put, offshore outsourcing significantly increases both types of risks as a company needs to manage information exchange with both its customer and its offshore vendor, effectively making itself to a middle person.

In terms of requirements risks, a company needs to gather accurate, adequate functional and quality requirements and ensure that all requirements are correctly interpreted by its offshore
outsourcing vendor. In a typical scenario without an offshore vendor, a company gathers requirements from its customers and attains an accurate interpretation of requirements by iteratively consulting with its customer throughout a product development lifecycle. A relatively straightforward two-way communication is no longer feasible with offshore outsourcing as a company is no longer capable of assessing the accuracy and adequacy of requirements, at least not in an effective, timely manner, because its offshore vendor is responsible for designing and development based on given requirements. Furthermore, significant geographical and temporal distance between a company and its offshore outsourcing vendor poses an additional layer of challenges in conveying ongoing changes in product requirements initiated by customers.

Given that a company retains an offshore outsourcing vendor to implement software solutions, be it software design and development or just the latter, the company essentially loses a direct control over implementation risks faced by a specific project. The situation is particularly acute when a company is inexperienced in engaging offshore outsourcing for its software PDD, and possesses no prior knowledge on the technical competencies of its chosen outsourcing vendor. Under this circumstance, it is crucial that a company does its due diligence ensuring its vendor possesses the technical competencies to delivery and is able to meet deliverables through a product development lifecycle. The technical competencies refer to a vendor’s ownership of knowledge required for designing and developing a software product that meets all requirements provided by its customer. The value proposition of a vendor is greatly diminished if it needs to acquire of knowledge not yet owned during a project lifecycle.
Effective management of requirements risks and implementations risks can enable a company to create a comprehensive approach for its PDD strategy and implementation. As implied throughout this section, communication is extremely important as a company must manage information flow and exchange with its customer and offshore outsourcing vendor. Management of risks requirements, if done properly, can lead to an accurate project scoping and avoid scope creep, and hence a project plan with realistic scheduling and precise cost estimates. Similarly, proper management of implementations risks can enable a customer to monitor a project progress closely, and detect implementation setbacks promptly throughout a product development lifecycle.

4.4 Analysis of Project Under Study

The project under study is analyzed using the integrative implementation framework proposed in Section 4.3. The analysis of the project focuses on root causes and events around the key issues identified by the survey responses in Chapter 3. Table 7 entails the tabulation of five key issues. The analysis takes place in a sequence according to the proposed implementation framework, starting with the specified aspects of general management and proceeding to the specified aspects of engineering. The specified aspects of general management layer are corporate strategy and business decision making, and organization processes and structure. The engineering layer comprises systems and project management, and product design and development.

<table>
<thead>
<tr>
<th>Key Issues Identified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Project Rationale and Engagement of Product Outsourcing Unclear:</td>
</tr>
<tr>
<td>Business decisions on product re-architecting and engagement of offshore outsourcing vendor in application development were considered unclear by the project team.</td>
</tr>
</tbody>
</table>
### Table 7 - Key Issues Identified on Project under Study

<table>
<thead>
<tr>
<th>4.4.1 General Management Layer – Project Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>4.4.1.1 Corporate strategy &amp; Business Decision Making:</strong></td>
</tr>
<tr>
<td>- GBTL did not seem to have a comprehensive outsourcing strategy. The main goal of the project is to re-architect the POS system at lowest costs. The offshore outsourcing vendor for the project was selected from a group of bidders solely based on the cost factor. GBTL’s outsourcing strategy did not explicitly take into account the impacts of the business decision in terms of dependency for knowledge versus dependency for capacity as proposed in Figure 15. In terms of the evaluation of outsourcing risks as proposed in Figure 16, the risks seem to fall into both quadrants A and B, where the client-tier is decomposable and possesses a clearly defined interface and GBTL’s dependency on its vendor for both knowledge and capacity. Notwithstanding, the program manager considered it was a good decision to outsource because “the portion outsourced can always be brought back in house” (Section 7.1.2.1 – Response to Question 6).</td>
</tr>
</tbody>
</table>
retrospect, due diligence could have been done on the evaluation of potential vendors in important areas, such as their track records in application development and competencies in specific technical domains required by the project. A comprehensive outsourcing strategy should look beyond lowering costs and strive for maintaining the company’s competitive edge by sustaining and strengthening intellectual and technical skills. Instead of confining its outsourcing strategy to only a specific product, namely the POS system, GBTL should have planned for harnessing new engineering knowledge from the project and leveraging it across multiple products.

GBTL did not put in place a systematic way to assess the project’s feasibility for offshore outsourcing. It was very possible that the decision making process of the company was unaware of the difference between operational function outsourcing and application development outsourcing. As previously pointed out in Section 4.3.3, offshore outsourcing of application development introduces new challenges in addition to existing challenges typically facing offshore outsourcing of operational functions. Using the diagnostic framework for offshore outsourcing of application development in Figure 17, GBTL could have been able to assess the feasibility of the project under study for offshore outsourcing. The diagnostic framework points to a low level of “instability of business requirements” and a medium to high level of “numbers of interfaces to other system.” POS 4.x is a distributed application system with a high numbers of interfaces among major components in the three-tier architecture as well as the number external interfaces with customer’s backend system. The horizontal dimension on interactions requirements points to a medium level in both customer input and internal interaction. The majority of product requirements come from a major customer and the level of interaction among project team members is generally
moderate when compared to other products. The assessment leads to a diagnosis that the feasibility of the project for outsourcing falls into the zone between category 2 and category 4. The diagnosis is very useful for requiring the company to put in place high interaction handling capabilities and tools and adjust its project plan to accommodate a longer transition. Figure 19 illustrates that POS 4.x is moderately suitable for offshore outsourcing provided that GBTL understands the underlying challenges and put in place appropriate management mechanism and tools for handling these challenges.

![Figure 19 - Feasibility of POS 4.x for Offshore Outsourcing](image)

4.4.1.2 Organization Structure and Processes

- Offshore outsourcing of application development requires an organization design that facilitates information exchange and flow and implements effective management of
information gained. The project team encountered major organizational challenges working with the offshore outsourcing vendor where it was plagued with ineffective information exchange and flow. Prior to the engagement of the vendor in the product development efforts, the project team was structured a matrix organization, where each team member belonged to a functional group and a product group. The negative effects of these challenges were particularly acute in the beginning of the project. As Figure 7 in Section 2.4.4 illustrates the projection team organizational structure of POS 4.x, the information exchange and flow between the internal project team members and the vendor took place in a highly dispersed manner. Each internal functional group exchanged information with their vendor counterparts without any formal involvement of other functional groups. There were virtually no defined processes provisioned for managing design knowledge and propagating design decisions to all functional groups in the project team. For instance, product design decisions were made by system architects and the vendor without adequate, timely input from internal functional groups. Further exacerbated by System Architects’ lack of product knowledge on the POS system, several key design decisions resulted in the product being overly architected.

- As discussed in Section 4.3.2, it is crucial that the design of organization structure and processes focuses on enabling a globally dispersed and temporally separated project team to communicate effectively and manage information knowledge accordingly. Before the organizational design process can take place, the company must understand its existing organization structure and processes, and ensure that effective information exchange and flow is currently taking place under the given
structure. Following the first step is the assessment of potential impacts of an offshore vendor on the information exchange and flow. The given design process, if conducted diligently, can lead to a project team organization structure with well-defined processes that enable effective knowledge management and information exchange and flow.

- **Figure 20** shows a viable example of project team organization design for the project under study. The example closely resembles the actual organization structure and processes after the original organization design was altered by the program manager who was brought in to take over the project management. In this instance, the program manager played a lead role in making effective changes to the organization structure and processes, and enabling information exchange and flow among internal functional groups and between the internal project team and the offshore outsourcing vendor. Meetings were held on a regular basis to formalize information exchange and propagate information flow to all functional groups in the least time consuming way. A knowledge repository system for the project was created using an existing information management system to store all documentations and meeting minutes for easy access by all project team members. The organization design essentially enabled effective communication without creating excessive red-tapped procedures. Such organization structure and processes were likely to have enabled effective communication for the project from the very beginning had the project team organization design were properly created and put in place.
4.4.2 Engineering Layer – Project Analysis

4.4.2.1 Systems and Project Management

- Inappropriate SPM methods were used for the project under study. The analysis of the project characteristics points to inadequate information as the project involved the implementation of new technology and GBTL was inexperienced in managing offshore outsourcing of application development. Hence, the paper considers the “learning” strategy as the most suitable SPM strategy for the POS 4.x project. The project manager responsible for creating the project plan was only familiar with projects associated with product upgrades of POS 3.x. Software upgrade projects typically face no technical uncertainties as the technology remains relatively constant from one product upgrade to
the next upgrade. By default, the project under study was incorrectly managed using of
the “instructionism” strategy as delineated in Table 5. Applying inappropriate SPM
strategy for the project resulted in the failure of the project plan to account for
uncertainties of unknown-unknowns posed by inadequate information. The mistaken use
of a sequential approach in identifying and arranging project tasks, followed by fatal
tries to arrange these tasks into parallel tasks resulted in pervasive schedule slippage.
The project manager was oblivious to the learning process that needed to take place. The
use of “learning” strategy would have eased up the required learning by breaking down
the project into tasks that could be arranged in iterative ways, such as spiral PM
approach, to improve project planning through learning over time. In the same token,
“selectionism” strategy also facilitates the required learning process for effective
handling of inadequate information by a pursuit of a set-based approaching leading to the
selection of the best alternative. Given that the “selectionism” strategy is considerably
more costly to deploy, the paper views the “learning” strategy the most appropriate view
for the project under study.

- According to the study of the general master plan for POS4.0 (Appendix A – Section
6.1), the project plan failed to apply any control mechanism for monitoring the project
progress. The failure to monitor the project progress in a timely manner precluded timely
detections of and prompt reactions to schedule slippage. The failure to instill the project
team’s capability to closely monitor the project progress can be overcome by the use of
time-based control, such as deadlines and milestones. The failure resulted in significant
schedule slippage and cost overrun, and the termination of the original project manager.
A seasoned program manager was brought in from external hiring to manage the project.
The program manager abandoned the “instructionism” SPM strategy and replaced it with the “learning SPM strategy, where he reconstructed the project plan into multiple release phases. Equally important, as illustrated in the general master plan for POS 4.1 (Appendix A – Section 6.3), project milestones were installed as time-based control for monitoring the project progress closely. The installation of time-based control is useful in promoting greater cross functional collaboration as the project team strives for meeting interim deadlines set for project milestones.

4.4.2.2 Product Design and Development

- The survey findings suggest that the project under study failed in the management of both requirement risks and implementation risks. In the management of requirement risks, the project focused mostly the functional requirements and completely overlooked the quality requirements. Despite a biased toward focusing only requirements risks, the project encountered major design errors – first, the client-tier was overly architected; second, customer’s rejection of database redesign for achieving better performance and greater scalability. In the management of implementation risks, the mistaken application of SPM approach for POS 3.x projects indiscriminately to POS 4.x project resulted in a failure to recognize the presence of implementation risks in the project plan. Due diligence was also missing in verifying that the chosen offshore outsourcing vendor did possess the technical skills to develop the GUI component of POS 4.x.

- The analysis viewed System Architects’ lack of product knowledge as the leading cause of an overly architected client-tier. The project relied on the Technical Analyst as the main source of the functional requirements and assigned product design tasks to System
Architects who lacked in-depth product knowledge in POS 3.x. Based on the analysis, the design flaw is mainly due to two conflicting functional requirements on the client component. First requirement called for a browser-based thin client to take advantage of more efficient software deployment and reuse of existing hardware at physical store locations. A browser-based thin client requires inevitable changes to user interface for minimizing the frequency of data transmission. Second requirement called for minimal end-user training by preserving the existing look and feel of the user interface. It can be achieved by adopting a thick design with trade-off in a greater system resource usage. The specified trade-off was significantly exacerbated when the System Architects did not object to VERT’s design decision on implementing Java InfoBus internal data exchange.†

The focus on fulfilling the second requirement without compensating for the first requirement led to an “overly architected” client-tier.

- The inadequacy of GBTL in managing functional requirements also led to customer’s rejection of the database redesign and forced the project team to discard the implementation and revert to the existing database design. The rejection was extremely costly and defeated a major objective of product re-architecting, namely a new product architecture that provides better performance and greater scalability. The key customer rejected the redesign after it became aware of the impending obsolescence of its internal applications that used the POS database as data source. In this aspect, GBTL was unaware of the key customer “hooking” into the existing POS database via its internal

---

* The thick client design was commonly referred as Java Swing client because it consisted of building graphical user interfaces using Swing classes.

† The InfoBus is a standard extension API consisting of the javax.infobus package. It is an API that defines Java classes and interface for data exchange operations. The InfoBus is provisioned for enabling data exchange between Java objects using Producer/Consumer model.
applications for business purposes unrelated to the POS product functionality itself, and therefore did not fully engage the customer in understanding the potential ramifications of database redesign during the design phase.

- The project did a particularly poor job in managing quality requirements. Almost all areas of quality requirements were completely overlooked by the project. The project oversimplified the requirements to a general statement that the POS 4.x product will behave and perform the same if not better than the POS 3.x product. The failure gave rise to many undefined acceptance criteria for quality requirements. In other words, the project failed to define quality criteria, such as reliability, performance, scalability, and security, rendered extremely frustrating and confusing for engineers to work toward meeting quality requirements without quantitative targets. In short, the project’s failure to manage quality requirements contributed greatly to serious underestimations of its schedule, costs, and allocation of engineering resources.

- Given that GBTL decided to utilize offshore outsourcing to reduce its product development costs and selected its vendor solely on the basis of costs, the paper argues that the company was responsible for managing implementation risks. In the selection of vendor, GBTL failed to detect that its chosen offshore vendor did not have the required technical skills to develop assigned product components until setbacks emerged during system integrations. To make things worse, the ability to mitigate implementation risks was hampered by its ineffective SPM strategy. Deploying the “instructionism” strategy, though the project required the “learning” strategy, precluded the project from handling uncertainties posed by inadequate information. In this specific project, the sources of
implementation risks can be traced to the inadequate outsourcing strategy and the mistaken SPM strategy.

- The project under study also exhibited a fundamental incompetence in managing system integration efforts. Again, the source of the problem can be traced to original project manager who was inexperienced in management new product development. Inadequate experience in this critical area led the project manager to pursue the aggressive delivery date set by the upper management. In the attempts of the project manager to strive for the delivery date, futile attempts were made to rearrange sequential development tasks into parallel tasks. The outcome created major challenges in integrating various components. The key to ensure feasible system integration efforts on software components constructed in a parallel is to break down a system into discreet, independent units with clear defined interfaces. The development of an enterprise software product is mostly about effective system integration efforts but not about programming tasks.

4.5 Chapter Summary

The chapter emphasizes that the importance of context specificity in establishing the proposed implementation framework for software product re-architecting. Offshore outsourcing of application development apparently adds an additional layer of complexity to software product development efforts. With the utilization of the proposed implementation framework, the analysis validates the interrelations among corporate strategy on offshore outsourcing, organization design, systems and project management, and product design and development. In the area of corporate strategy on offshore outsourcing, the analysis of the project under study points to the importance of outsourcing strategy, evaluation of the
feasibility of a project for offshore outsourcing, and selection of appropriate offshore vendors. Organization design exhibits the importance of managing the challenges of information flow introduced by a geographically dispersed and temporally separated project team. Systems and project management is equally important as it provides an effective framework for managing offshore outsourcing projects based on the level of information availability. The realm of project design and development points out the need for managing both requirements risks and implementation risks facing software engineering in general. In short, the chapter shows that offshore outsourcing of application development, while appears to be a plausible way for lowering costs, requires a coherent, integrative approach to ensure its successes. Given the management complexity facing the offshore outsourcing of application development, the author is unsure on the abilities of most US companies to effectively engage offshore outsourcing vendor in their software project development efforts.
This page is intentionally blank.
5 Chapter 5: Conclusion

5.1 Thesis Summary

The author was unable to locate any existing case studies on software project failure with characteristics similar to those of the project under study. It is likely most real-world cases of unsuccessful software development projects are downplayed by companies, especially those pertain to the offshore outsourcing of application development. During the data collection phase, the author found that data on project setbacks were harder to come by than those on project successes. The case study was made possible with the author’s direct involvement in the project under study and his former colleagues’ willingness to participate in written surveys and follow-up interviews.

The case study evolved around three distinct characteristics of the project under study. The re-architecting of an established software product engendered two distinct characteristics – the constancy of functional requirements and the implementation of new technology. The third distinct characteristic was the offshore outsourcing of application development where GBTL decided to engage an offshore outsourcing vendor in its product re-architecting efforts. The constancy of functional requirements led key decision-makers to establish project delivery dates that many project team members viewed as overly aggressive. Key decision-makers made a conscious business decision to pursue offshore outsourcing as the means for minimizing project costs and implementing technology that was new to GBTL.

The case study attributed most problems encountered in the areas of systems and project management (SPM) and product design and development (PDD) to improper handling of key
issues emerged from the project's three distinct characteristics. A concise summary of the SPM and PDD issues is as follows:

- **Issues related to the constancy of functional requirements:**
  
The project under study contained no clearly defined quality requirements and implementation requirements. The original master project plan focused almost exclusively on its functional requirements and essentially overlooked the relevance of quality requirements and implementation requirements. The execution of the original project plan failed to account for the quality and implementation attributes of the re-architected product. The oversight was likely due to GBTL’s inherent organizational strengths in product maintenance and upgrades and relatively inconsistent track records on new product creation. As a result, the project did not proactively manage uncertainties/risks with missing requirements in the area of quality and implementation. Inadequate handling issues in this specific area resulted in significant schedule slippage and major setbacks, such as customer’s rejection of the database-tier implementation of database-tier and missing key implementation elements.

- **Issues related to the implementation of new technology**
  
The original master project plan was created without accounting for uncertainties/risks inherent in the implementation of new technology. Instead of applying SPM approaches that address information inadequacy and facilitate the acquisition of new technical knowledge, the project was managed in ways similar to that of a product upgrade, where a project plan contained groupings of sequential tasks, and was based on an incorrect assumption of adequate information. Attempts to target the delivery dates established by key decision makers led to parallel groupings of sequential tasks that complicated system
integration efforts. The inadequate handling of issues related to this project characteristic was partially responsible for the mishandling of a design trade-off between two conflicting functional requirements on the client component.

- **Issues related to the offshore outsourcing of application development**

GBTL did not appear to have a carefully thought-out outsourcing strategy. It did not anticipate complexities arising from its engagement of an offshore outsourcing vendor in its product development efforts. As a result, it did not evaluate the needs for modifying its internal organization structure and processes. Although GBTL would depend on its outsourcing vendor for the implementation of new technology, it selected its vendor solely on the basis of costs. The vendor turned out not to have the technical expertise it purported to possess. The lack of the required technical expertise was responsible for the client-tier design that was “overly architected.”

The case study concludes that the offshore outsourcing of application development requires not less, as it has been widely perceived, but more management efforts. First and foremost, it is very important to recognize that offshore outsourcing of application development does not have exactly the same attributes as other simpler forms of offshore outsourcing arrangements. Effective offshore outsourcing of application requires a carefully thought out outsourcing strategy and companies should not pursue it solely for the purpose of cost reductions. In the case study, while offshore outsourcing did not account for every major setback encountered by the project, it had certainly created an additional layer of SPM and PDD complexity that required management attentions. Only a few of the project setbacks can be attributed to a single cause or issues within a project characteristic. Instead, the majority
of the setbacks arose from the dynamic interaction of issues within all three distinct project characteristics.

5.2 SDM Prescriptions from Lessons Learned

For software re-architecting projects of similar characteristics, the paper recommends an integrative approach according to the lessons learned from the case study:

- **Outsourcing Strategy & “Offshorability” Assessment:**
  Companies need to have a carefully thought out outsourcing strategy with valid reasons to pursue the outsourcing venue. Their reasons should be more than just reducing development costs and include maintaining competitive edge and acquiring new knowledge. For each project under consideration, companies need to understand whether it incurs knowledge dependency or capacity dependency on their offshore outsourcing vendors (Figure 16 - Matrix of Dependency and Outsourcing). Accordingly, companies need to assess the “offshorability” of application development based on the management requirements and interaction requirements of a project (Figure 17 - Diagnostic Framework: Offshore Outsourcing of Application Development).

- **Organization Design - Information Flow and Knowledge Management:**
  Companies must put in place organization structure and processes that facilitate collaboration and information exchange. They must beware that offshore outsourcing further increases the collaboration intensiveness among team members that are geographically and temporally separated. Companies need to ensure that their organizations support more than just informal information exchange and ad hoc knowledge management. Companies are required to view its organization structure and
processes as an underlying infrastructure for their implementation of their SPM and PDD processes.

- **Systems and Project Management (SPM) – Level of Information Adequacy:**
  Companies must evaluate the level of information adequacy of a project in an accurate manner. The evaluation of information adequacy enables companies to identify and manage uncertainties in specific project elements, including those assigned to their offshore outsourcing vendors. Also, an accurate assessment on information adequacy enables companies to consider applying recommended key methods (Table 5 - Three Fundamental SPM Strategies). Equally important, time-based control, such as milestones and interim deadlines, is useful for measuring project progress on a timely basis. Companies should always ensure that their SPM methods and tools are compatible with their organization designs and software development approaches.

- **Product Design and Development (PDD) - Risk Management:**
  Assuming that companies are proficient in software design and development tools, the key to effective PDD lies in managing project uncertainties inherent in software engineering. Offshore outsourcing tends to magnify project uncertainties. Project uncertainties are due to requirements risks and implementation risks (Table 6 - Summary of PDD (Software Engineering) Risks). Companies must put equal emphases on both functional and quality requirements. Viewing software development as the application of knowledge owned or to be acquired, companies can manage implementations risks by determining what knowledge is required for fulfilling product requirements.
This page is intentionally left blank.
### 6 Appendix A

#### 6.1 Project Schedule - General Master Project Plan for POS 4.0

The following table is the high-level project schedule derived from the original master project schedule signed off by the management team of the project under study. The ownership of tasks is grouped into two categories, where the tasks in green cells belong to the outsourcing vendor and yellow cells to the company of the project.

<table>
<thead>
<tr>
<th>#</th>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RMS Project</td>
<td>197 days</td>
<td>Mon 12/10/01</td>
<td>Fri 9/13/02</td>
</tr>
<tr>
<td>2</td>
<td>RMS Project Inception</td>
<td>9 days</td>
<td>Mon 12/10/01</td>
<td>Thu 12/20/01</td>
</tr>
<tr>
<td>3</td>
<td>Project Kickoff Activities</td>
<td>9 days</td>
<td>Mon 12/10/01</td>
<td>Thu 12/20/01</td>
</tr>
<tr>
<td>14</td>
<td>Infrastructure</td>
<td>5 days</td>
<td>Mon 12/10/01</td>
<td>Fri 12/14/01</td>
</tr>
<tr>
<td>21</td>
<td>Inception Deliverables</td>
<td>0 days</td>
<td>Thu 12/20/01</td>
<td>Thu 12/20/01</td>
</tr>
<tr>
<td>32</td>
<td>RMS Project Elaboration</td>
<td>61.50 days</td>
<td>Wed 12/19/01</td>
<td>Wed 3/20/02</td>
</tr>
<tr>
<td>33</td>
<td>Requirements</td>
<td>8 days</td>
<td>Wed 12/19/01</td>
<td>Thu 1/3/02</td>
</tr>
<tr>
<td>34</td>
<td>Functional/Non-Functional Requirements</td>
<td>7.50 days</td>
<td>Wed 12/19/01</td>
<td>Thu 1/3/02</td>
</tr>
<tr>
<td>40</td>
<td>Project Plan (1.0)</td>
<td>4 days</td>
<td>Thu 12/27/01</td>
<td>Thu 1/3/02</td>
</tr>
<tr>
<td>44</td>
<td>Test Plan</td>
<td>8 days</td>
<td>Wed 12/19/01</td>
<td>Thu 1/3/02</td>
</tr>
<tr>
<td>51</td>
<td>Analysis and Design</td>
<td>12 days</td>
<td>Fri 12/21/01</td>
<td>Fri 1/11/02</td>
</tr>
<tr>
<td>52</td>
<td>Design</td>
<td>12 days</td>
<td>Fri 12/21/01</td>
<td>Fri 1/11/02</td>
</tr>
<tr>
<td>59</td>
<td>Update Project Plan (2.0)</td>
<td>6 days</td>
<td>Fri 1/4/02</td>
<td>Fri 1/11/02</td>
</tr>
<tr>
<td>63</td>
<td>Acceptance Criteria</td>
<td>3.50 days</td>
<td>Fri 3/15/02</td>
<td>Wed 3/20/02</td>
</tr>
<tr>
<td>68</td>
<td>RMS Project Construction</td>
<td>178 days</td>
<td>Wed 1/2/02</td>
<td>Thu 9/5/02</td>
</tr>
<tr>
<td>69</td>
<td>RMS Beta-1</td>
<td>103 days</td>
<td>Wed 1/2/02</td>
<td>Fri 5/24/02</td>
</tr>
<tr>
<td>70</td>
<td>RMS Beta-1 Project Plan Refinement</td>
<td>2.50 days</td>
<td>Fri 3/22/02</td>
<td>Tue 3/26/02</td>
</tr>
<tr>
<td>74</td>
<td>RMS Beta-1 Elaboration</td>
<td>103 days</td>
<td>Wed 1/2/02</td>
<td>Fri 5/24/02</td>
</tr>
<tr>
<td>75</td>
<td>Database Store Procedure Development</td>
<td>8 days</td>
<td>Fri 2/22/02</td>
<td>Tue 3/5/02</td>
</tr>
<tr>
<td>79</td>
<td>Create Flow Charts for Feature List</td>
<td>8 days</td>
<td>Wed 1/9/02</td>
<td>Fri 1/18/02</td>
</tr>
<tr>
<td>80</td>
<td>RMS Beta-1 Technical Design</td>
<td>32 days</td>
<td>Fri 2/15/02</td>
<td>Mon 4/1/02</td>
</tr>
<tr>
<td>121</td>
<td>RMS Beta-1 Test Plan/Test Cases</td>
<td>50 days</td>
<td>Thu 2/14/02</td>
<td>Wed 4/24/02</td>
</tr>
<tr>
<td>132</td>
<td>RMS Beta-1 Test Script Development</td>
<td>103 days</td>
<td>Wed 1/2/02</td>
<td>Fri 5/24/02</td>
</tr>
<tr>
<td>139</td>
<td>RMS Beta-1 Construction</td>
<td>45 days</td>
<td>Mon 3/25/02</td>
<td>Fri 5/24/02</td>
</tr>
<tr>
<td>150</td>
<td>RMS Beta-1 Final Release to Lightbridge</td>
<td>0 days</td>
<td>Fri 5/24/02</td>
<td>Fri 5/24/02</td>
</tr>
<tr>
<td>151</td>
<td>Feedback on RMS Beta-1</td>
<td>5 days</td>
<td>Fri 5/24/02</td>
<td>Thu 5/30/02</td>
</tr>
<tr>
<td>152</td>
<td>RMS Beta-1 Defect Prioritization / Assignment</td>
<td>1 day</td>
<td>Wed 5/29/02</td>
<td>Wed 5/29/02</td>
</tr>
<tr>
<td>153</td>
<td>RMS Beta-1 Signoff</td>
<td>0 days</td>
<td>Fri 5/31/02</td>
<td>Fri 5/31/02</td>
</tr>
<tr>
<td>154</td>
<td>RMS Beta-2</td>
<td>104 days</td>
<td>Thu 3/14/02</td>
<td>Wed 8/7/02</td>
</tr>
<tr>
<td>155</td>
<td>RMS Beta-2 Project Plan Refinement</td>
<td>8 days</td>
<td>Mon 5/6/02</td>
<td>Wed 5/15/02</td>
</tr>
<tr>
<td>159</td>
<td>RMS Beta-2 Elaboration</td>
<td>85 days</td>
<td>Thu 3/14/02</td>
<td>Thu 7/11/02</td>
</tr>
<tr>
<td>160</td>
<td>Beta 2 Database/Store Procedure Development</td>
<td>72 days</td>
<td>Thu 3/14/02</td>
<td>Fri 6/21/02</td>
</tr>
<tr>
<td>170</td>
<td>RMS Beta 2 Workflow Design</td>
<td>36 days</td>
<td>Wed 4/3/02</td>
<td>Wed 5/22/02</td>
</tr>
<tr>
<td>178</td>
<td>RMS Beta-2 Technical Design</td>
<td>58 days</td>
<td>Mon 4/15/02</td>
<td>Wed 7/3/02</td>
</tr>
<tr>
<td>Task Description</td>
<td>Duration</td>
<td>Start Date</td>
<td>End Date</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------</td>
<td>------------</td>
<td>----------</td>
<td></td>
</tr>
<tr>
<td>RMS Beta-2 Test Plan/Test Cases</td>
<td>33 days</td>
<td>Mon 5/27/02</td>
<td>Thu 7/11/02</td>
<td></td>
</tr>
<tr>
<td>RMS Beta-2 Construction</td>
<td>52 days</td>
<td>Mon 5/27/02</td>
<td>Wed 8/7/02</td>
<td></td>
</tr>
<tr>
<td>Lightbridge Development Tasks</td>
<td>0 days</td>
<td>Fri 6/14/02</td>
<td>Fri 6/14/02</td>
<td></td>
</tr>
<tr>
<td>Beta 2 Drop 1</td>
<td>25 days</td>
<td>Mon 5/27/02</td>
<td>Fri 6/28/02</td>
<td></td>
</tr>
<tr>
<td>Beta 2 Drop 2</td>
<td>38 days</td>
<td>Fri 6/14/02</td>
<td>Wed 8/7/02</td>
<td></td>
</tr>
<tr>
<td>Beta 2 Reports Development</td>
<td>40 days</td>
<td>Mon 5/27/02</td>
<td>Fri 7/19/02</td>
<td></td>
</tr>
<tr>
<td>Beta 2 Test Script Development</td>
<td>50 days</td>
<td>Mon 5/27/02</td>
<td>Fri 8/2/02</td>
<td></td>
</tr>
<tr>
<td>Beta-2 Defect Prioritization / Assignment</td>
<td>2.5 days</td>
<td>Wed 7/31/02</td>
<td>Fri 8/2/02</td>
<td></td>
</tr>
<tr>
<td>RMS Final Beta-2 Signoff</td>
<td>0 days</td>
<td>Fri 8/9/02</td>
<td>Fri 8/9/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1</td>
<td>73 days</td>
<td>Thu 5/23/02</td>
<td>Sat 8/31/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release-1 Project Plan refinement</td>
<td>3 days</td>
<td>Mon 7/29/02</td>
<td>Wed 7/31/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Elaboration</td>
<td>69 days</td>
<td>Thu 5/23/02</td>
<td>Tue 8/27/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Database/Store Procedure</td>
<td>60 days</td>
<td>Fri 5/31/02</td>
<td>Thu 8/22/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Workflow Design</td>
<td>11 days</td>
<td>Thu 5/23/02</td>
<td>Thu 6/6/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Technical Design</td>
<td>34 days</td>
<td>Sun 6/23/02</td>
<td>Thu 8/8/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Test Plan/Test Cases</td>
<td>25 days</td>
<td>Wed 7/24/02</td>
<td>Tue 8/27/02</td>
<td></td>
</tr>
<tr>
<td>RMS Release 1 Construction</td>
<td>29 days</td>
<td>Wed 7/24/02</td>
<td>Sat 8/31/02</td>
<td></td>
</tr>
<tr>
<td>Release 1 Development</td>
<td>29 days</td>
<td>Wed 7/24/02</td>
<td>Sat 8/31/02</td>
<td></td>
</tr>
<tr>
<td>Release 1 Test Script Development</td>
<td>21 days</td>
<td>Thu 8/1/02</td>
<td>Fri 8/30/02</td>
<td></td>
</tr>
<tr>
<td>Release 1 Defect Prioritization / Assignment</td>
<td>5 days</td>
<td>Sat 8/31/02</td>
<td>Thu 9/5/02</td>
<td></td>
</tr>
<tr>
<td>Transition</td>
<td>6 days</td>
<td>Fri 9/6/02</td>
<td>Fri 9/13/02</td>
<td></td>
</tr>
</tbody>
</table>
6.3 Project Schedule - General Master Project Plan for POS 4.1

The following table is the high-level project schedule derived from the master project schedule provisioned by the new Program Manager. The project plan was specifically devised to bring the project under study back on track.

<table>
<thead>
<tr>
<th>#</th>
<th>Task Name</th>
<th>Days</th>
<th>Start</th>
<th>Finish</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Development Milestone Schedule (Program's Plan)</td>
<td>84 days</td>
<td>Fri 11/22/02</td>
<td>Thu 3/27/03</td>
</tr>
<tr>
<td>2</td>
<td>M1 - Stable POS 4.1 build with reverted POS 4.1 Client’s DB</td>
<td>0 days</td>
<td>Fri 11/22/02</td>
<td>Fri 11/22/02</td>
</tr>
<tr>
<td>3</td>
<td>M2 - Clean drop for QA SPR verification and preliminary testing</td>
<td>0 days</td>
<td>Fri 12/6/02</td>
<td>Fri 12/6/02</td>
</tr>
<tr>
<td>4</td>
<td>M3 - Build 5 drop for QA SPR verification and preliminary testing</td>
<td>0 days</td>
<td>Fri 12/20/02</td>
<td>Fri 12/20/02</td>
</tr>
<tr>
<td>5</td>
<td>M4 - Quality/Pre-release to QA</td>
<td>0 days</td>
<td>Fri 1/3/03</td>
<td>Fri 1/3/03</td>
</tr>
<tr>
<td>6</td>
<td>M5 - Quality Official Release to QA</td>
<td>0 days</td>
<td>Mon 1/20/03</td>
<td>Mon 1/20/03</td>
</tr>
<tr>
<td>7</td>
<td>M6 - Quality/Product</td>
<td>0 days</td>
<td>Thu 3/27/03</td>
<td>Thu 3/27/03</td>
</tr>
<tr>
<td>8</td>
<td>Development Milestone Completion</td>
<td>0 days</td>
<td>Thu 3/27/03</td>
<td>Thu 3/27/03</td>
</tr>
<tr>
<td>10</td>
<td>Internal Technical Design</td>
<td>103 days</td>
<td>Mon 8/12/02</td>
<td>Wed 1/8/03</td>
</tr>
<tr>
<td>11</td>
<td>Write Technical Specs &amp; Review</td>
<td>103 days</td>
<td>Mon 8/12/02</td>
<td>Wed 1/8/03</td>
</tr>
<tr>
<td>12</td>
<td>Secured Functions</td>
<td>15 days</td>
<td>Mon 8/12/02</td>
<td>Fri 8/30/02</td>
</tr>
<tr>
<td>16</td>
<td>Signature Capture Changes</td>
<td>4 days</td>
<td>Mon 9/9/02</td>
<td>Thu 9/12/02</td>
</tr>
<tr>
<td>20</td>
<td>Electronic Check Authorization (ECA)</td>
<td>16 days</td>
<td>Mon 8/12/02</td>
<td>Tue 9/3/02</td>
</tr>
<tr>
<td>26</td>
<td>Customer Maintenance</td>
<td>15 days</td>
<td>Mon 8/12/02</td>
<td>Fri 8/30/02</td>
</tr>
<tr>
<td>31</td>
<td>3G Enhancement</td>
<td>14 days</td>
<td>Tue 8/13/02</td>
<td>Fri 8/30/02</td>
</tr>
<tr>
<td>36</td>
<td>Sales Receipts (Front Office &amp; Back Office)</td>
<td>16 days</td>
<td>Thu 9/12/02</td>
<td>Thu 10/3/02</td>
</tr>
<tr>
<td>42</td>
<td>Reporting Tech Spec</td>
<td>18 days</td>
<td>Mon 8/12/02</td>
<td>Thu 9/5/02</td>
</tr>
<tr>
<td>47</td>
<td>Airtime Payments (ATP)</td>
<td>8 days</td>
<td>Wed 10/23/02</td>
<td>Fri 11/1/02</td>
</tr>
<tr>
<td>52</td>
<td>EPX</td>
<td>13 days</td>
<td>Tue 11/19/02</td>
<td>Mon 12/9/02</td>
</tr>
<tr>
<td>58</td>
<td>WebStart</td>
<td>6 days</td>
<td>Mon 11/25/02</td>
<td>Wed 12/4/02</td>
</tr>
<tr>
<td>61</td>
<td>Datamart</td>
<td>7 days</td>
<td>Mon 11/25/02</td>
<td>Thu 12/5/02</td>
</tr>
<tr>
<td>64</td>
<td>InstallAnywhere Spec</td>
<td>17 days</td>
<td>Fri 12/13/02</td>
<td>Wed 1/8/03</td>
</tr>
<tr>
<td>67</td>
<td>Write Tech Specs &amp; Review Complete</td>
<td>0 days</td>
<td>Wed 10/2/02</td>
<td>Wed 10/2/02</td>
</tr>
<tr>
<td>68</td>
<td>Update Tech Specs based on DB Changes Complete</td>
<td>5.50 days</td>
<td>Tue 9/24/02</td>
<td>Tue 10/1/02</td>
</tr>
<tr>
<td>75</td>
<td>Update Tech Specs based on DB Changes Complete</td>
<td>0 days</td>
<td>Tue 10/1/02</td>
<td>Tue 10/1/02</td>
</tr>
<tr>
<td>76</td>
<td>Technical Design Complete</td>
<td>0 days</td>
<td>Wed 1/8/03</td>
<td>Wed 1/8/03</td>
</tr>
<tr>
<td>78</td>
<td>Receive 4.0 Final</td>
<td>0 days</td>
<td>Fri 8/30/02</td>
<td>Fri 8/30/02</td>
</tr>
<tr>
<td>80</td>
<td>DB Engineering Setup</td>
<td>18 days</td>
<td>Wed 8/14/02</td>
<td>Mon 9/9/02</td>
</tr>
<tr>
<td>86</td>
<td>DB Engineering Setup Complete</td>
<td>0 days</td>
<td>Mon 9/9/02</td>
<td>Mon 9/9/02</td>
</tr>
<tr>
<td>88</td>
<td>Rolled Back DB for 4.1</td>
<td>0 days</td>
<td>Mon 11/4/02</td>
<td>Mon 11/4/02</td>
</tr>
<tr>
<td>90</td>
<td>Development</td>
<td>120 days</td>
<td>Wed 9/4/02</td>
<td>Tue 2/25/03</td>
</tr>
<tr>
<td>91</td>
<td>Post 4.0 Work</td>
<td>13 days</td>
<td>Tue 9/17/02</td>
<td>Thu 10/3/02</td>
</tr>
<tr>
<td>93</td>
<td>Signature Capture Changes</td>
<td>17.60 days</td>
<td>Fri 9/27/02</td>
<td>Tue 10/22/02</td>
</tr>
<tr>
<td>Task Description</td>
<td>Duration</td>
<td>Start Date</td>
<td>End Date</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>-----------</td>
<td>---------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>Electronic Check Authorization (ECA)</td>
<td>76 days</td>
<td>Wed 9/4/02</td>
<td>Fri 12/20/02</td>
<td></td>
</tr>
<tr>
<td>Sales Receipts (Front Office &amp; Back Office)</td>
<td>6.90 days</td>
<td>Mon 10/14/02</td>
<td>Tue 10/22/02</td>
<td></td>
</tr>
<tr>
<td>Reporting</td>
<td>32.95 days</td>
<td>Thu 9/5/02</td>
<td>Mon 10/21/02</td>
<td></td>
</tr>
<tr>
<td>EPX Enhancements</td>
<td>20 days</td>
<td>Wed 12/11/02</td>
<td>Thu 1/9/03</td>
<td></td>
</tr>
<tr>
<td>WebStart</td>
<td>28 days</td>
<td>Mon 12/2/02</td>
<td>Fri 1/10/03</td>
<td></td>
</tr>
<tr>
<td>Datamart</td>
<td>60 days</td>
<td>Wed 11/27/02</td>
<td>Tue 2/25/03</td>
<td></td>
</tr>
<tr>
<td>InstallAnywhere Development</td>
<td>3 days</td>
<td>Fri 12/13/02</td>
<td>Tue 12/17/02</td>
<td></td>
</tr>
<tr>
<td>Development of Enhancements Complete</td>
<td>0 days</td>
<td>Thu 1/31/02</td>
<td>Thu 1/31/02</td>
<td></td>
</tr>
<tr>
<td>Milestone 4 QA Build - Integration testing</td>
<td>0.50 days</td>
<td>Fri 1/3/03</td>
<td>Fri 1/3/03</td>
<td></td>
</tr>
<tr>
<td>Testing InstallAnywhere with a Q/A deployment</td>
<td>0.50 days</td>
<td>Fri 1/10/03</td>
<td>Fri 1/10/03</td>
<td></td>
</tr>
<tr>
<td>Document Automated Nightly Build - Mini-Doc</td>
<td>9 days</td>
<td>Mon 12/30/02</td>
<td>Fri 1/10/03</td>
<td></td>
</tr>
<tr>
<td>Automated nightly build and regression tests</td>
<td>11 days</td>
<td>Thu 12/19/02</td>
<td>Fri 1/10/03</td>
<td></td>
</tr>
<tr>
<td>Nightly Regression Testing</td>
<td>5 days</td>
<td>Mon 1/13/03</td>
<td>Fri 1/17/03</td>
<td></td>
</tr>
<tr>
<td>Integration Testing Phase</td>
<td>5 days</td>
<td>Mon 1/13/03</td>
<td>Fri 1/17/03</td>
<td></td>
</tr>
<tr>
<td>QA Tasks</td>
<td>58.50 days</td>
<td>Fri 1/3/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>Pre-QA in Parallel with IT Phase I</td>
<td>11 days</td>
<td>Fri 1/3/03</td>
<td>Mon 1/20/03</td>
<td></td>
</tr>
<tr>
<td>Official QA</td>
<td>48 days</td>
<td>Mon 1/20/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>QA Complete</td>
<td>0 days</td>
<td>Thu 3/27/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>DB Tasks</td>
<td>64.50 days</td>
<td>Tue 12/24/02</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>Upgrade Scripts</td>
<td>7 days</td>
<td>Fri 12/27/02</td>
<td>Tue 1/7/03</td>
<td></td>
</tr>
<tr>
<td>DB Setup for Pre-QA Build</td>
<td>6 days</td>
<td>Tue 12/24/02</td>
<td>Fri 1/3/03</td>
<td></td>
</tr>
<tr>
<td>DB Setup for Full QA</td>
<td>5 days</td>
<td>Tue 1/14/03</td>
<td>Mon 1/20/03</td>
<td></td>
</tr>
<tr>
<td>DB Tasks for Client Prep</td>
<td>5 days</td>
<td>Fri 3/21/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>DB Tasks Complete</td>
<td>0 days</td>
<td>Thu 3/27/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>Performance Engineering</td>
<td>40 days</td>
<td>Tue 2/4/03</td>
<td>Tue 4/1/03</td>
<td></td>
</tr>
<tr>
<td>Create New Text Scripts</td>
<td>15 days</td>
<td>Tue 2/4/03</td>
<td>Tue 2/25/03</td>
<td></td>
</tr>
<tr>
<td>Performance Testing</td>
<td>22 days</td>
<td>Wed 2/26/03</td>
<td>Thu 3/27/03</td>
<td></td>
</tr>
<tr>
<td>Write Formal Disclosure Report</td>
<td>3 days</td>
<td>Fri 3/28/03</td>
<td>Tue 4/1/03</td>
<td></td>
</tr>
<tr>
<td>Performance Engineering Complete</td>
<td>0 days</td>
<td>Tue 4/1/03</td>
<td>Tue 4/1/03</td>
<td></td>
</tr>
<tr>
<td>Additional Tasks &amp; Durations T-B-D prior to GA</td>
<td>332 days</td>
<td>Thu 1/31/02</td>
<td>Wed 5/21/03</td>
<td></td>
</tr>
<tr>
<td>Documentation</td>
<td>1 day</td>
<td>Thu 1/31/02</td>
<td>Thu 1/31/02</td>
<td></td>
</tr>
<tr>
<td>Training</td>
<td>1 day</td>
<td>Thu 1/31/02</td>
<td>Thu 1/31/02</td>
<td></td>
</tr>
<tr>
<td>Client UAT</td>
<td>30 days</td>
<td>Wed 4/2/03</td>
<td>Tue 5/13/03</td>
<td></td>
</tr>
<tr>
<td>GA Prep</td>
<td>5 days</td>
<td>Wed 5/14/03</td>
<td>Tue 5/20/03</td>
<td></td>
</tr>
<tr>
<td>GA</td>
<td>1 day</td>
<td>Wed 5/21/03</td>
<td>Wed 5/21/03</td>
<td></td>
</tr>
</tbody>
</table>
This page is intentionally left blank.
7 Appendix B

7.1 Survey Responses

The appendix is provisioned for the repository of all survey responses to the general questionnaire and the transcript of a one-on-one interview with the Program Manager responsible for the project under study at GBTL. Note that the author has filtered out business sensitive issues related to the company (TBGL) and individuals of the project under study according to the confidentiality agreement of the survey.

7.1.1 Written Responses

The section contains the actual survey output from the survey respondents. Each of the survey responses is mapped to a respondent code as listed in Table 8. The author would likely to acknowledge his peers for their willingness to share their experiences in and professional opinions on the project under investigation. Based on the verbal agreements reached in the beginning of this term project, the detailed information of the organization, project specifics, and names of participants will not be disclosed to public.

<table>
<thead>
<tr>
<th>Role of Respondent</th>
<th>Respondent Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Architect (Replacement)</td>
<td>SA</td>
</tr>
<tr>
<td>Software (Database) Architect</td>
<td>DA</td>
</tr>
<tr>
<td>Technical Analyst</td>
<td>TA</td>
</tr>
<tr>
<td>Documentation Specialist</td>
<td>DS</td>
</tr>
<tr>
<td>Database Administrator</td>
<td>DBA</td>
</tr>
<tr>
<td>Technical Consultant</td>
<td>TC</td>
</tr>
<tr>
<td>QA Manager</td>
<td>QA</td>
</tr>
</tbody>
</table>

Table 8 - Role of Survey Respondents

7.1.1.1 Respondent Code: SA

1) Do you consider the project a success or a failure? Why?
We learned a lot of thing from this project, and I think the company gained something out of it. I consider the project a failure, not because how good or bad the code was developed, but from the point of view of why did we do it, and the basic requirements for the re-architecting task.

2) What areas went wrong? What was your reaction if applicable?
As I just said, the starting point of the project was wrong:
- Why did we want to re-architect it, considering the requirement is to duplicate the exact functionalities and look-and-feel?
- What is the business reason, benefits, or customer base for the task?

Other than the management failures, there were problems in the architecture design: the solution was over architected. A lot of the problems we had later (bad code, poor performance, etc.) were related to the over architect problem. Another thing did not went well at first was the way we work with offshore contractors. I won't call it a total failure, since we did a better job later on from our earlier failures.

3) What areas went right? What was your reaction if applicable?
It is hard to find anything positive as the project was over shadowed by tons of problems. But we did some things correctly. For example, we kept some importance features in house: performance, database, etc. We did a lot better later on with offshore contractor folks as we learned along the way.

4) What should the project team do differently if it could work on the same project again?
One thing is to re-examine the requirements more carefully even before any technical work. We could also do a better job in communication of the architect and design (more formal documentation, presentations, workshops, etc.) not only between the company and offshore contractor, but also among our own team. Get more GBTL team members involved as early as possible, and give them more responsibilities. You cannot have two architects come in make a design, and as the work is in full progress, they just went away.

5) What's your role in the project?
Technical lead.

7.1.1.2 Respondent Code: DA

1) Do you consider the project a success or a failure? Why?
I would say that this project was a failure, from the simple definition that the product never shipped. The client is still using the 3.x generation of the product.

2) What areas went wrong? What was your reaction if applicable?
I was involved mostly at the initiation and early development stages of the product. There were a number of challenges, some that were faced for the first time by the company:
- This was the first project that used outsourcing. The Company had no experience with outsourcing and we did not manage the relationship with our outsource vendor well. The
original project manager for the outsourcing vendor had to be replaced because of "chemistry" issues.

- The in-house architects were not developers who had worked on the product before. Thus, they could make decisions only on the basis on technical expertise, not on product knowledge, which they had to learn as they went along.
- The development schedule was too aggressive. This was a major re-write of the product in a new technology. Further, feature development continued on the 3.x version of the product, which meant that the 4.x version was always going after a moving target.
- Some technology decisions were made without consulting with the customer. This contributed to the customer rejecting the final product.
- The client insisted that the database not change. This meant that attempts to standardize naming conventions, to improve performance at the database tier, and to take advantage of database re-factoring were lost.

3) What areas went right? What was your reaction if applicable?
On the plus side, significant functionality was delivered within a reasonable time. The technological hurdles were mostly overcome. The Company gained valuable experience dealing with outsourcing.

4) What should the project team do differently if it could work on the same project again?
Among other things, I would assemble a team that understood the business rules and the nature of the product before making technology decisions. It was questionable whether this product warranted the cost and effort put into the rewrite. A business case and ROI should have been prepared.

5) What's your role in the project?
I worked on the re-write of the database and PL/SQL stored procedures. My role was primarily architectural.

7.1.1.3 Respondent Code: TA *

1) Do you consider the project a success or a failure? Why?
The project was a success from the management perspective since it was time and resource bounded. But was a failure from client perspective due to the architectural design decision made without client's consent. The client was not comfortable with the design even though the re-architected product was functionally equivalent.

2) What areas went wrong? What was your reaction if applicable?
Here are few factors that went wrong:
- Lack of communication: Communicating the design to the client before the actual development.

* The survey response by the Technical Analyst appears to contradict the rest of the survey responses. Perhaps the contradiction was due his functionality role as the provider of product knowledge to the offshore contractor. Given that his responsibilities took in the upstream of software development lifecycle, it was unlikely that he was at the receiving end of engineering challenges posed by poor software quality and scalability.
Lack of prototype implementation: There was no phase in the project to develop and test a prototype based on the re-architected design.

3) What areas went right? What was your reaction if applicable?
The project management was good. The entire project was outsourced with limited supervision. Lot of documentation was developed along the way. The outsourced firm met the goal with extra effort. It was a good learning exercise for in-house engineering and consulting teams.

4) What should the project team do differently if it could work on the same project again?
- More communication and coordination with client
- Layout design and prototype before the actual development initiatives
- Look for alternative design and functional changes to meet the same requirements.

5) What's your role in the project?
I was acting as technical analyst for the conversion project. I was working closely with the outsourced firm in explaining how the existing product works. Also in translating code and knowledge from old architecture to the new architecture.

7.1.1.4 Respondent Code: DS
1) Do you consider the project a success or a failure? Why?
Failure. Never saw it run in production because it didn't meet client's needs.

2) What areas went wrong? What was your reaction if applicable?
- Poor leadership.
- Poor communication.
- Lack of involvement of all interested parties from the start.
- Contractors imposed their own methodology which was unfamiliar to us.
- Contract programmers didn't understand the product or the business. Also seemed to have no experience implementing an online help system and could not seem to follow direction when given (possible language or culture barrier).
- Management either didn't understand client acceptance thresholds or felt they could talk their way around them.

My reaction was to cover my butt - paper trails on all my issues. Also needed to escalate issues to my boss more frequently than in past, which resulted in her micromanaging me.

3) What areas went right? What was your reaction if applicable?
I got my work done, on schedule, and felt my deliverable was of better quality than the legacy system I'd been managing with for 3.x. Gave me a limited sense of accomplishment (would have been better to see it in production!).
4) What should the project team do differently if it could work on the same project again?
- Better planning.
- Better leadership.
- Better management of contractors.
- Better cross-functional communication.
- Better understanding/acceptance of limitations early on that could cause the deliverable to be unacceptable to the client (so we don't bother spending a year + to develop something that is too big and slow!)

5) What's your role in the project?
Documentation writer and designer.

7.1.1.5 Respondent Code: DBA

1) Do you consider the project a success or a failure? Why?
I consider the project a failure. Though there were pieces of it that succeeded, particularly the Specifications Phases, many people were unrealistic about what it would take to succeed at a project of this size. The deadlines were moved and changed, the metrics for success were undefined (other than to say that there would be no severity 1 bugs in the app, and the project manager kept deciding that certain things weren't severity 1).

2) What areas went wrong? What was your reaction if applicable?
There were so many areas that went wrong that it's difficult to pinpoint which were the worst. I think the main thing was the management of the project. They were more invested in looking good than actually producing a good product. It was really hard to watch.

I would say that my reaction was primarily frustration. Many other managers realized what was going on but, for some reason, upper management didn't handle the problem. The other thing that was hard was the interaction between offshore contractors and the employees.

There was a lack of confidence in the contractors' work that was not mitigated so there was a lot of finger pointing when things went wrong. Additionally, though I know it is a problem for many projects, the final date was set by our client and the functionality was set also. So there came a time when we were just throwing bodies at the project in the hopes that we could meet an unrealistic deadline.

3) What areas went right? What was your reaction if applicable?
The specification phase of the project went well. The technical requirements were well known and enough time was spent evaluating technical options in terms of the front end. This led to some good architecture decisions.*

The inconsistency of the remark with the survey findings is likely due to the respondent not joining the team until the last one third of the project life cycle. It's fair to assert design shake out was complete by the time the respondent was assigned to the project team.

111
4) What should the project team do differently if it could work on the same project again?
At the company, I am not sure there was much that could change but from a purely academic perspective, I would not hire offshore consultants unless there was much more of a team-building effort. The internal Java team could have done that work if their manager were required to get involved. Also, the project management team should have been more realistic about deadlines and goals and not moved the project's goals over time.

5) What's your role in the project?
I was the lead DBA for the second half of the project.

7.1.1.6 Respondent Code: TC
1) Do you consider the project a success or a failure? Why?
I consider the project a failure because it didn't produce anything useful. It didn't yield anything other than short term consulting revenues. The client isn't using the product.

2) What areas went wrong? What was your reaction if applicable?
- Scope creep causing resource and scheduling bloat
- Continuous and pervasive under-estimation of resources and schedule
- Miscommunication and spin through out the organization
- (mis-)Use of outsourcing
- Huge amount of SPR's (note: reported software defects)
- Unstable code base necessitated many revisions of QA and Performance evaluation cycles
- Serious lack of or very late in the game "production readiness"
- Little or no regard for the bottom line.

My reaction was one of pessimism and doubt. Moral was generally low, as were expectations of the outcome.

3) What areas went right? What was your reaction if applicable?
- Short term and reoccurring consulting revenue.
- Client management (I guess)
- Lots of "low hanging fruit" for performance evaluation

4) What should the project team do differently if it could to work on the project again?
- Fire itself and save the company the trouble
- Get the client to commit to the scope
- Work with the client to ensure that it is what they need

5) What's your role in the project?
My role was to provide moral support to the performance engineer (and occasional technical consulting).
7.1.1.7 Respondent Code: QA

1) Do you consider the project a success or a failure? Why?
Failure. The QA requirement to test was equal to or better than the legacy system built on PowerBuilder. The new POS 4x never hit that requirement.

2) What areas went wrong? What was your reaction if applicable?
- Planning: Due diligence of the proposed platform (Java) was not evaluated as to whether the product could perform well enough to be a viable solution.
- Also, we never had the requirements from the client nailed down prior to development start. We had to back pedal when the client changed dB schema half way through the project.
- Some powers never believed the product was worth investing in. Client was the major financial contributor.

My input fell on deaf ears. I just went along with the unrealistic expectations of this project.

3) What areas went right? What was your reaction if applicable?
Configuration Management (CM) and QA. Both the QA department and the CM department learned much from the experience. For the CM he learned a brand new build process of J2EE.

I just took it all in stride. I was fortunate to gain the experience of working with a third party development and QA organization from India and Sri Lanka.

4) What should the project team do differently if it could work on the same project again?
Plan better. Get the company to buy in from the top and force the clients input up front.

5) What's your role in the project?
QA Manager.

7.1.2 One-on-One Interview with Program Manager

The author analyzed the survey responses and tabulated results into categories of interesting insights. Under the condition that none of the business and personnel sensitive information would be disclosed, the Program Manager of the software project under study graciously agreed to a one-on-one interview with the author. As previously noted, the Program Manager

* Upon a follow-up clarification, the respondent meant to refer to customer’s objection to the database schema change. Just like the rest of the project team members, the respondent was under the impression that the customer had agreed to the database scheme redesign until it was subsequently objected. It was unclear where the ball was dropped, i.e. some blamed it on the company’s failure to obtain customer consent in writing during the design phase while some on the constant of management on the customer due to its continuous restructuring and cost cutting efforts.
was brought in via external hiring to replace the original Project Manager. The original Project Manager was terminated two months after the Program Manager joined the project team.

7.1.2.1 Interview Note

The interview lasted an hour and fifteen minutes and took place at the program manager’s office. The interviewing process entailed two set of questions, namely customized questionnaire and general questionnaire. The interview began with customized questions derived specifically from the tabulation and analysis of the responses to the general questionnaire survey. The interview was transcribed by the author in the following day after the interview.

7.1.2.2 Response to Customized Questionnaire

1. **What was the business rationale behind the project initiation?**
   Based on hearsay:
   - Primary customer wanted a more open architecture
   - Organization did not view existing PowerBuilder based client and client/server architecture as a competitive platform
   - Organization needed a new architecture that provide better customization capability

2. **You mentioned about risk mitigation several times. How important is risk mitigation as a program manager?**
   Risk mitigation is about “anticipation.” It is important to identify all the risk points and focus on the big risk points. The risk mitigation process is mostly based on intuition and experience. Based on risk factors/uncertainties, I put accountability where it is. It means forward thinking and anticipating what can go wrong.

3. **Could you provide me with the background information on the status of POS 4.x when you came aboard the team as the programming manager?**
   I came aboard in February 2002 with the mandate to decide if there is a go or no go with the offshore outsourcing provider. The provider stepped up its efforts by assigning experienced managers.

4. **Do you know why the original architects were taken off the team?**
   The decision was political.

* The Program Manager qualified the response as “hearsay” because he joined the project team in Feb 2003 and was not involved in the initially planning of the project.
5. Do you think outsourcing is appropriate for this project? What are your decision making criteria when it comes to outsourcing?
It was a good decision to outsource. It’s a business strategy. The portion outsourced can always be brought back in house. The GUI design was an over-engineered (as a side note).

6. What are critical skills in program management?
- PM requires a histogram of skills
- Ability to create “context” is key
- Technical skill
- Ability to identify what needs to be tightly managed – “long leash versus short leash”
- “Bull-shit meter” based upon personal experience to know when to probe deeper
- Know what you are capable of, i.e. “know when and where to get help”
- Leverage one’s core skill sets to manage amid ambiguities and abstraction.

7.1.2.3 Response to General Questionnaire

1) Do you consider the project a success or a failure? Why?
It’s a failure because the product did not ship on time and was never deployed in productions. It’s a success because the organization demonstrated that it could develop product in a different way, i.e. outsourcing. It learned a new type of project management and product management.

2) What areas went wrong? What was your reaction if applicable?
- Underestimation of the requirement efforts
- Training of existing staff – less than 50% crossed the technical chasm
- Bad design of GUI component (info bus) – re-factoring (for what??)
- Mismanagement of vendor (outsourcing provider) relationship

3) What areas went right? What was your reaction if applicable?
- Product delivery and fulfillment of customer contract (get paid in a low single digit of million dollars)†
- Learn valuable lessons from managing outsourcing activities
- Gain new technical knowledge – i.e. J2EE and Java Swing

4) What should the project team do differently if it could work on the same project again?
- Reduce touch points with the vendor

* There was an ongoing turf war and distrust between the Enterprise Architecture Group and Consulting Group since the latter took over the POS Software Development Group. The Directory of Enterprise Architecture pulled his System Architects from the project team after the project started to encounter imminent schedule slippage and blaming game was initiated primarily by the Consulting Group.
† The PM refused to disclose the details on the actual negotiation between GBTL and its primary wireless customer. The final outcome of the negotiation resulted in the transfer of source code ownership to the primary customer.
- Better manage the thick (info bus) versus thin client design and requirement
- Better manage the requirement of the database component (database schema change and SP rewrite) – the changed was rejected by customers at a product technical summit with the primary customer.

5) What's your role in the project?
Program Manager
8 Bibliography:


Campbell, B., (TBD), Real World Offshore Development Practices, PM Boulevard


Hayes, M. (2003), Precious Connection - Companies thinking about using offshore outsourcing need to consider more than just cost savings, *InformationWeek*, October 20, 2003


Shenhar, A., Dvir, D., Lechler, T., Poli, M. (NA), One Size Does Not Fit All - True For Projects, True For Frameworks, *Working Paper - TBD*, Steven Institute of Technology & Ben-Gurion University

Simmons, L. (2004), How to manage outsourcing risks, *ZDNet Australia*, October 27, 2004


This page is intentionally left blank.
9 Annotated References:

1 Shenhar, A., Dvir, D., Lechler, T., and Poli, M. (NA) One Size Does Not Fit All - True For Projects, True For Frameworks, Working Paper - TBD, Steven Institute of Technology & Ben-Gurion University


4 Hayes, M. (2003), Precious Connection - Companies thinking about using offshore outsourcing need to consider more than just cost savings, InformationWeek, October 20, 2003

5 Hayes


8 Fine et al.

9 Gupta et al.


11 Campbell, B. (NA), Real World Offshore Development Practices, PM Boulevard


14 Pich et al.


16 Pich et al.


