A MIGRATION MANAGEMENT FRAMEWORK PROPOSAL FOR COBOL/CICS BASED MAINFRAMES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HALİL KAPLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JANUARY 2010

Approval of the thesis:

A MIGRATION MANAGEMENT FRAMEWORK PROPOSAL FOR COBOL/CICS BASED MAINFRAMES

submitted by **HALİL KAPLAN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Müslim Bozyiğit _____
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Ali H. Dogru _____
Supervisor, **Computer Engineering Dept**., **METU**

Examining Committee Members

Assoc.Prof.Dr. Ahmet Cosar _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Ali H. Dogru _____
Computer Engineering Dept., METU

Dr. Cevat Şener _____
Computer Engineering Dept., METU

Dr. Sinan Kalkan _____
Computer Engineering Dept., METU

Taylan Güvercin _____
Industrial Engineer, MBA, SIEMENS

Date: **08.01.2010**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Halil KAPLAN

Signature :

ABSTRACT

A MIGRATION MANAGEMENT FRAMEWORK PROPOSAL

FOR COBOL/CICS BASED MAINFRAMES

KAPLAN, HALİL

M. S., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Ali Hikmet Doğru

January 2010, 129 pages

Today, mainframes contain a considerable portion of business applications worldwide. It is estimated that the current inventory of production COBOL running on mainframes is 150 to 200 billion lines of code. Despite the efforts to change within the mainframe environment, these mainframes, nowadays, face major problems for host organizations due to a combined set of financial, business related, technical and organizational problems. Moreover, the factors like diminishing resources (COBOL programmers), lack of documentation, inability to integrate with other systems, increasing maintenance costs, etc. have caused the organizations search for migration solutions. To overcome this problem within the context of modernization, over the years several main migration approaches that ranges from simple screen scraping methods to complete re-write of applications or re-hosting of platforms have been developed.

To contribute to the solution of this overall problem, this thesis proposes a methodology framework specifically for the COBOL/CICS based mainframes. The research studies in this topic within this field are mainly focused on the technical aspects whereas our concentration is covering not only that but the other

essential aspects of the problem domain. These are organizational view, project management view and process view. Within the thesis study, a special interest is given to the modernization strategy selection among migration, rewrite, packaged and do-nothing alternatives. Experimental results are also provided within the thesis to prove the usability of the approach for this selection.

Keywords: Mainframe Migration, Migration Framework, Modernization Strategy Selection, Legacy Migration

ÖZ

COBOL/CICS TABANLI BÜYÜK SİSTEMLERE YÖNELİK TAŞIMA YÖNETİM ÇERÇEVESİ ÖNERİSİ

KAPLAN HALİL

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Danışman: Doç. Dr. Ali Hikmet Doğru

Ocak 2010, 129 sayfa

Günümüzde, dünya genelindeki büyük sistemler iş uygulamalarının önemli bir bölümünü barındırmaktadır. Halen büyük sistemlerde çalışmakta olan COBOL envanterinin 150 ila 200 milyar kod satırı olduğu tahmin öngörülmektedir. Büyük sistemlerin kendi ortamı içerisindeki değişim çabalarına rağmen büyük sistemler, günümüz itibariyle içinde bulundukları kuruluşlar için toplamda çok ciddi finansal, iş bazlı, teknik ve organizasyonel problemlere yol açmaktadırlar. İlaveten, kaynaklardaki daralma (COBOL programcıları), dokümantasyon eksiklikleri, diğer sistemlerle entegrasyon zafiyetleri, artan bakım masrafları v.b. faktörler nedeniyle ilgili kuruluşlar modernizasyon kapsamında, taşıma (migration) çözümleri arayışına girmişlerdir. Bu problemin giderilebilmesi amacıyla yıllar içerisinde, basit ekran kopyalama metotlarından uygulamaların tümüyle yeniden yazılması veya platformun yeni ortamda barındırılması gibi geniş bir yelpazede yaklaşımlar geliştirilmiştir.

Bu genel problemin çözümüne katkıda bulunabilmek amacıyla, bu tez çalışması ile COBOL/CICS tabanlı büyük sistemlere yönelik olarak bir metodoloji çerçevesi önerisi sunulmaktadır. Bu alandaki araştırmalar genelde

teknik bakış alanında yoğunlaşmakla beraber, bu çalışma teknik boyuta ilave olarak problem alanının kaçınılmaz diğer temel boyutlarını da ele almaktadır. Bunlar organizasyonel boyut, proje yönetim boyutu ve süreç boyutlarıdır. Bu tez çalışması içerisinde; taşıma, yeniden yazma, paket uygulamalara geçiş ve bir şey yapmama alternatifleri içerisinden modernizasyon stratejisinin seçimi konusuna özel bir önem verilmiştir. Bu seçim yaklaşımının kullanılabilirliğini ispat amacıyla deneysel sonuçlara da tez içerisinde yer verilmiştir.

Anahtar Kelimeler: Büyük Sistemlerin Taşıması, Taşıma Çerçevesi, Modernizasyon Stratejisi Seçimi, Özgün Sistem Taşıması

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisor, Assoc. Prof. Dr. Ali Doğru, for his calm guidance, support and encouragement throughout my thesis work. I am also grateful for their initial support to Dr. Meltem Turhan Yöndem and Dr. Onur Tolga Şehitoğlu.

Lastly, I would like to thank my family and my colleagues Ahmet İzzet Erçevik and Burcu Kural for their support during my thesis study.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

The goal of this thesis study is the development of a migration management framework which is intended for migration of mainframes or so called legacy systems. As a prerequisite to the migration, it is also aimed to develop a decision model for the selection of appropriate modernization strategy among the alternatives of migration, re-write, packaged applications and do-nothing.

Mainframe migration has been an area of research and application mainly especially after 1990 with an increasing trend. Similarly due to the mainframe related problems faced by all organizations with mainframes, it is among the top priority problems that need to be handled. A recent Gartner worldwide survey among 1527 CIOs indicates that "Legacy application modernization" is ranked as fourth among the "Top 10 Business and Technology Priorities in 2009" and in 2009, more than 57 percent of CIOs reported this as one of their top five business expectations [1].

Table 1 Top 10 Business and Technology Priorities in 2009

Top 10 Business Priorities	**Rank**	**Top 10 Technology Priorities**	**Rank**
Business process improvement	1	Business intelligence	1
Reducing enterprise costs	2	Enterp.apps.(ERP, CRM, others)	2

Table 1 (continued)

Improving enterprise workforce effectiveness	3	Servers and storage technologies (virtualization)	3
Attracting and retaining new customers	4	Legacy application modernization	4
Increasing the use of information/analytics	5	Collaboration technologies	5
Creating new products or services (innovation)	6	Networking, voice and data communications	6
Targeting customers and markets more effectively	7	Technical infrastructure	7
Managing change initiatives	8	Security technologies	8
Expanding current customer relationships	9	Service-oriented applications and architecture	9
Expanding into new markets and geographies	10	Document management	10

International Data Corp. [2] estimates that 200 billion lines of legacy codes are still in use today on more than 10.000 large mainframe sites.

To understand the motivation of this thesis, first I would like explain why there is a motivation to migrate from mainframes to modern platforms as one of the modernization alternatives.

Legacy systems typically form the backbone of information flow within an organization and are normally mission-critical; a failure in one of these systems may have a serious business impact. They contain business rules and workflow controls that help organizations process orders, track revenues, manage inventories and perform other key business functions. Thus, for many organizations, decommissioning these systems is not an option. Operating and maintaining these mainframes constantly cause the following typical problems:

- Risks associated with running potentially unsupported HW and SW.

- Lack of uniformity and flexibility to add new products and services that contemporary platforms offer to meet the changing business needs.

- Technical obsolescence, hence, lack of integration and web capabilities for different business applications.

- Lack of competencies (due to retirement) to maintain and operate mainframe systems.

- High costs associated with maintaining and upgrading these systems. Tracing failures is costly and time consuming typically due to lack of documentation and a general lack of understanding of the internal workings of the systems after many years from the original development.

- Inability to evolve to provide new functionality in acceptable productivity and time ranges required for their organization to remain competitive.

- Inability to integrate. Integration efforts are greatly hampered by the absence of clean interfaces due to their monolithic nature [3]. These mainframes which are typically over 20 years old were architected in times when the integration was not a high priority concern and the main intent was to work by itself with a strict set of rules, processes and user interfaces.

Overall, these mainframe systems present a high level of entropy: the source code has become ill-structured, poorly documented, and weakly modeled [3]. So the motivation for the mainframers to relieve from or to get rid of these technical, business or organizational problems through migration or modernization efforts is obvious.

Within this thesis study, a two-staged approach is given to attack the problems stated above:

1. Firstly, the problem with the selection of the appropriate modernization strategy is handled. Before selecting the right

approach for modernization, first of all whether the migration is the right modernization strategy is to be determined. A decision model is proposed for this decision with all the criteria. For each organization, depending on the various technical and non-technical parameters, the appropriate strategy may be different among migration, packaged application, rewrite and do-nothing alternatives.

2. Secondly, for the case with selection of migration alternative; a migration management framework is proposed to handle the migration project.

The migration approaches to relieve from the stated problems have been developed especially within the last decade which had started with academic studies and followed by the industrialized solutions and migration tools. These migration approaches ranges from simple UI level modernization solutions like screen scraping, or adding web user interfaces, to complete re-engineering solutions, language conversion or application rehosting solutions in the new target (typically Windows or Linux based) platforms.

Regardless of the variety of these approaches, obviously there is no “silver bullet” type of a solution that is appropriate for all cases. Realistically, switching off a twenty or thirty year old system and plugging in a new feature rich replacement overnight is not an easy or realistic option. Similarly, attempting to recover the business logic and constructing replacements is also not an easy task which brings together many risks to be handled carefully. The organization will eventually face what is called the legacy dilemma [3] [4] [5]:

- It is expensive and risky to replace the legacy system
- It is even more expensive to maintain the legacy system.

To help understand the size of the problem, consider the program levels described in Frederick Brooks’ classic book "The Mythical Man Month" [6] Brooks claimed that there are 3 steps to reach a well functioning, mature application set: A Program, a Programming Product, and finally a Programming System Product. Applications often start as programs written quickly by a couple of skilled developers, possibly as prototypes. They are then adopted into the long-term application suite and development process which requires greater testing,

formal documentation and maintenance at which point they have progressed to the "Programming Product" stage. Then, over time, their interfaces are enriched; they are integrated with other systems and are given many detailed enhancements, finally reaching the status of "Programming System Product". Brooks asserts that you have to spend 3 times more effort in going from one of these steps to a higher step, so that you have to spend at least 9 times more effort to reach a Programming System Product from a Program. However, when we estimate the amount of work required to recreate an application that is at the Programming System Product level we are inclined to think of it in simpler terms than it has attained and underestimate the effort in elevating an application from one stage to the next. So, the first caution is to say that either in writing from scratch or trying to pick out pieces of an existing application will involve more work than will at first appear.

Therefore, the replacement of legacy information systems is much more extensive than a straightforward project and such a project could, quite legitimately, address the areas of reverse engineering, business reengineering, schema mapping and translation, data transformation, application development, human computer-interaction and testing.

Among those, 1991 DARWIN project was one of the early projects aimed to perform a migration through an incremental approach, so-called "Chicken Little Methodology" which was proposed by Brodie and Stonebaker [7] [8].

After those initial attempts, the mainframers learned it the hard way with the failure of many migration projects due to underestimating the complexity of the problem. In the mean time, there has also been some positive progress in this field like the emergence of automation tools, conversion tools that were generated through the ideas or results of academic studies of many researchers and efforts of IT industry.

But, as we explained above, there is no one single solution that can be the cure for all types of organizations [9] and considering the complexity of the problem, a modernization strategy selection has to be studied thoroughly for each case. As the problem has many varying components (technical, business wise, administrative, etc), each organization (with business priorities, technical infrastructure, availability of technical staff and their competencies, financial considerations etc) is unique itself and an analysis needs to be carried out detailing

all these steps in order to determine the right modernization strategy for the organization.

On the other hand, despite the availability of several main mainframe migration approaches, there is still no agreed or widely accepted taxonomy [9] of these approaches. Similarly, there is not also a widely accepted terminology in this field for these mentioned migration approaches.

1.1. Objective Of The Thesis

In light of the descriptions above, it is the identified that a migration management framework for the mainframe migration projects is essential in this area. Considering all the factors influencing the success of migration projects, such a migration management framework with a well-defined set of phases and processes would be a very critical contribution in the field.

On this topic, there are few methodology proposals [10]. Also, there are some methodology or life cycle proposals from the migration tool vendors on the IT industry side. But, these proposals are mainly centered around the proposed or developed migration tools or concepts. Therefore, they are heavily dependent on the specific tools or the main focus is limited to technical part of the problem domain.

To contribute to the solution of this overall problem, this thesis proposes a methodology framework specifically for the COBOL/CICS based mainframes.

There are several categories of solution for legacy software; almost all the focus within the software engineering has been on technical aspects [7]. In consideration of this, the concentration in this study is covering not only that but the other essential aspects of the problem domain. Among others, most critical of those views are organizational view, project management view and process view. During the thesis study, it was observed that, despite the common misconception, the mainframe migration is not an ongoing activity but rather a change project. The affect of change shall be taken into account within the following domains: business process, organization, location, applications, technology and data. And based on this analysis with all stakeholders, based on the framework, the project stages shall be well-planned and implemented at all phases, from the initial assessment to the final deployment.

Within the proposed methodology, the phases of the typical migration project are drilled down to activities and sub-activities with clear definitions of purpose, input, output and methods. With this approach, the aim is to reach to a general (in the context of not being dependent to a specific tooling or vendor approach) framework that can be used with the well defined process for the targeted COBOL/CICS based mainframes' migration. Uniqueness of the thesis comes from the fact that it is targeting specifically the COBOL/CICS based mainframes

Throughout the thesis, a special importance is given to the "modernization strategy selection" process within the methodology. Because it is crucial to the success of the superseding migration project activities set out in the project phases of the methodology. As described above; in the absence of a well established terminology and taxonomy of migration approaches, an extensive analysis and research of available approaches are comparatively reviewed and analyzed within this thesis study. Due to the nature of the problem domain, the selection of the modernization strategy is not only a subject of pure technical domain but also deserves to be evaluated with a holistic view. The selected migration approach, in addition to being the technically right approach, it shall also be the right approach from other perspectives like organizational willingness, ability to address business needs of organization, integration requirements, competitive positioning, cost considerations, alignment with the overall organizations IT strategy etc.

1.2. Organization Of The Thesis

As described above the main focus of the thesis is to propose a methodology covering also a life cycle model for the migration of COBOL/CICS based mainframes. Beyond this introductory chapter, the thesis is organized as follows: In Chapter 2, necessary background on the concepts of modernization and migration, brief history of migration and literature review with the current status are given. A comparative analysis of modernization strategies among migrate, re-write, packaged and "do-nothing" alternatives is given in Chapter 3. Chapter 4 gives a blueprint of the proposed migration management framework. In chapter 5, the framework is presented in detail. Chapter 6 presents a conclusion, information on work conducted and discuss about future work.

CHAPTER 2

BACKGROUND

2.1. Mainframe Migration

In order to better define the problem domain of mainframe migration, it is important to start firstly with the definition of mainframe in the literature. Mainframes or legacy systems are defined in different terms depending on the context they are viewed.

The originators of mainframes [11] define a mainframe defined as "a computer system designed to continuously run very large, mixed workloads at high levels of utilization meeting user defined service level objectives."

From the mainframe migration or modernization view, one of the commonly accepted definitions of the mainframes is that these are systems that significantly resist modification and evolution to meet new and constantly changing business requirements [12]. Delving more into this definition, mainframes are further categorized as follows:

- A system of poor quality that resists change, regardless of the platform on which it runs;
- Any system, regardless of its quality, that runs on a proprietary or obsolete platform; or

- A system of poor quality that resists change and runs on a proprietary or obsolete platform [13].

Within this context, another definition of Legacy Systems is provided by NASCIO (National Association of State Chief Information Officers) in a Survey held in 2008 with State CIOs [14] as follows:

"A legacy system is not solely defined by the age of IT systems (e.g. 20 years) as there are many systems that were designed for continued upgrades, but the term also focuses on elements such as "supportability", "risk" and "agility", including the availability of software and hardware support, and the ability to acquire either internal or outsourced staffing, equipment or technical support for the system in question. The term may also describe the system's inability to adequately support "line-of-business" requirements or meet expectations for use of modern technologies, such as workflow, instant messaging (IM) and user interface."

From the definition, out of the survey conducted within USA State CIOs [14], among the seven criteria, "Inability to be adequately supported, maintained, or enhanced" was chosen by 82.8%, "Inability to meet business needs or system not agile enough to continually meet the challenging needs of the organization" was chosen by 79.3% and thirdly "obsolete hardware or software components" was chosen by 79.3% of participants as key characteristics of the mainframe.

Due to the nature of this definition, mainframe migration is principally concerned with systems that pose problems in technical, financial or administrative aspects. Therefore, the ultimate aim of the migration is the safe, risk-free, and rapid migration of a mainframe system to an open platform.

Within this effort, it is also critical to preserve the organization's assets wherever possible, and the elimination of technical risk to the organization by eliminating its dependence on proprietary or obsolete technologies [13].

Within this thesis study, we are referring to "modernization" as the high level strategical effort to overcome the stated problems with the mainframe, i.e., the alternative modernization options as given in Chapter 1, can be one of the following:

- Do-nothing
- Re-write
- Packaged Applications
- Migration

Once this "modernization strategy" selection is made, a further categorization is available under each. For the case with migration, all the sub categories are referred as "Migration Approaches" within this document.

With this broad definition of migration which includes the preservation or re-use of existing assets like architecture, application or data of existing system, the migration is closely related and covers the areas of re-engineering, information modeling, program analysis, reverse engineering, software comprehension, software visualization architecture recovery and data/database modeling. In the following section of literature review, these related fields are also analyzed.

Within this thesis study, a two-staged approach is given to attack the problem:

- Firstly, the problem with the selection of the appropriate "modernization strategy" is handled. Migration is one of those modernization alternatives. Before selecting the right approach for migration, first of all, whether the migration is the right modernization strategy is to be determined. A decision model is proposed for this decision with all the criteria.
- Secondly, if the chosen modernization option is migration, than a migration management framework is proposed to handle the migration project with the chosen migration approach.

The modernization strategy selection alternatives are categorized under 4 basic categories. This broader categorization is mainly based on the options available in front of organizations with a mainframe system. Due to the reasons stated in Chapter 1, organizations are firstly evaluating and looking for correct the strategy to solve their problems on strategy level before thinking of specific migration approaches available. Therefore, first, a categorization of modernization

strategy alternatives is given below before delving into specific migration approaches. These four main categories of "modernization strategy" are defined as follows:

1. Do Nothing: Although from a technical perspective, there are many technological alternatives for migration, other than the technological considerations, "Do nothing" is also a viable option based on the organization's status.

2. Packaged Application: Commercial of the shelf (COTS) application option involves moving completely to a standard package in the new environment like the standard ERP applications.

3. Rewrite: This option is mainly based on the complete or partial rewrite of the applications. This option can be handled either by re-using the business logic available within the existing applications. Therefore within this option it is critical on how the business logic extraction from these programs is handled.

4. Migrate: Among all the strategies, the decision to migrate is a checkpoint for further categorization. Once this alternative is chosen, the further categorization of the migration approach like language conversion, re-hosting, wrapping etc is done based on the existing and target environment and the underlying technological components and migration tools.

Within this thesis study, mainly two fundamental contributions are provided. The first one is the proposal for the selection of the correct modernization strategy and the second one is the framework offered for the migration projects which is not dependent on a specific vendor, technology or migration toolkit for the whole life cycle of a typical migration project.

For the first one, the problem is clearly stated by [15] that "there is no adequate decision method to help organizations decide on the investment strategy for the legacy systems". In Chapter 3 a detailed comparative analysis of the modernization strategies is provided.

For the second contribution topic, a brief history of migration together with a state of the art literature review of migration approaches inline with the proposed migration management framework is provided in the following section.

2.2. Literature Review

2.2.1. Taxonomy

In the academia or industry there is no single widely accepted definition of migration approaches. Moreover, there is no single accepted taxonomy or standard on grouping of migration approaches. This fact is stated in many different reports and researches. For example, Ovum report [9], as a message to the industry, indicates this :

"Legacy renewal is a growing but fragmented industry. There is no universally accepted taxonomy or methodology; terms like 'migration' can have multiple meanings depending on context and vendor. You must establish industry-wide standards for legacy renewal by active participation in industry groups, such as the OMG (see http://adm.omg.org/ – the Architecture-Driven Modernization taskforce); the new Open SOA Collaboration project (see http://www.osoa.org) and the Mainframe Migration Alliance (see www.mainframemigration.org)."

Among those taxonomies, an initial categorization was done by Weiderman and these activities were mainly divided into three main categories: maintenance, modernization and replacement [16]:

- Maintenance: Maintenance is an incremental and iterative process in which small changes are made to the system. These changes are often bug corrections or small functional enhancements and should never involve major structural changes.

- Replacement is appropriate for legacy systems that can not keep pace with business needs and for which modernization is not possible or cost effective. Replacement is normally used with systems that are undocumented, outdated, or not extensible.

- Modernization involves more extensive changes than maintenance, but conserves a significant portion of the existing system. These

changes often include system restructuring, important functional enhancements, or new software attributes. Modernization is used when a legacy system requires more pervasive changes than those possible during maintenance, but it still has business value that must be preserved.

Another approach is to categorize the approaches as White-Box and Black-box approaches based on requirement of the knowledge of the internals of a mainframe [17]: "Modernization that requires the knowledge of the internals of the legacy system is called white-box modernization, and modernization that just requires knowledge of the external interfaces of a legacy system is called black-box modernization."

Another taxonomy is provided by Erradi et al. [18] that the approaches are classified as non-evasive and evasive. This is based on the level of effort put in the migration. For example legacy wrapping, UI webification type of approaches which require less effort which provides only short term solutions are regarded as non-invasive, and on the other hand, componentization, reengineering or re-hosting alternatives are regarded as evasive approaches which required deep analysis of existing assets for long term solutions.

Due to the lack of a standard taxonomy, within this thesis study, "Modernization" is referred as the high level strategical initiative and decision among the high level alternative options of Do-nothing, Re-write, Packaged Applications and Migration. The "Migration" is one of those options and under this option there are various migration implementation categories which are referred as "Migration approaches" through out the thesis study.

2.2.2. A Brief History Of Migration Approaches

Aside from the taxonomy of migration approaches, the early migration approaches were mainly separated on being all-at-once (big bang approach) or incremental. The big bang approach (so-called "Cold Turkey" approach) [7] proposed to redevelop the legacy system from scratch, using a modern architecture, tools and databases, running on a new hardware platform. The main criticism for this approach was on the high risk of failure associated with any large software project. The other arguments against this approach were listed as follows [13]:

- Management rarely approves a major expenditure if the only result is lower maintenance costs, instead of additional business functionality.

- Development of such massive systems takes years, so unintended business processes will have to be added to keep pace with the changing business climate, increasing the risk of failure.

- Documentation for the old system is frequently inadequate.

- Like most large projects, the development process will take longer than planned, testing management's patience.

- And finally, there's a tendency for large projects to end up costing much more than anticipated.

So, against this big bang approach, with the DARWIN project, an incremental approach referred as “Chicken Little” was proposed and applied by University of Berkeley [7]. However, this incremental approach brought up another problem that is the need for gateways for the inter-operation and data exchange between the two systems during this increment process. The need for gateways added another layer of difficulty and complexity to an already complex problem and projects.

Ganti and Brayman proposed an approach in the form of guidelines to migrate to a distributed environment [8]. However, the approach was lacking sufficient details on the most critical phase of cutover from existing system to the target environment.

Another approach was proposed by [19] as part of MILESTONE project which was called “Butterfly Methodology”. It was also an incremental approach. The main difference of this one from “Chicken Little” was that there was no interop stage envisioned [19]:

“Different from Chicken little, the Butterfly methodology eliminates, during the migration, the need for system users to simultaneously access both the legacy and target systems, and therefore, eliminates the need of interoperation between these two (heterogeneous) information systems. It is very important to bear in mind

that, using the Butterfly methodology, the target system will not be in production while the legacy system is being migrated. The legacy system will remain in full production during the whole migration process. There will never be a case where live data is stored, at the same time, in both the legacy and target systems. Legacy system migration can be a very expensive procedure, which carries a definite risk of failure. In order to perform a successful migration, a sound model of the migration process is obviously needed. Currently, however, no general model exists. MILESTONE'S considers that migration consists of five major tasks:

1) Justification;

2) Legacy System Understanding;

3) Target System Development;

4) Migration;

5) Testing."

Although a wide consensus was available for the incremental approach [20-25], one important component was missing in early large-scale project trials. That is the lack of automated migration tools until mid 1990s. This was one the main reasons for the failure of many migration projects or limiting the migration projects only with small or medium sized organizations with limited MIPS counts and application sizes. Typically, there are thousands of applications and corresponding database entities, screens and system level utilities, without considerable automation it was not possible or too costly and error prone trying to handle these migration projects.

Although there have been many successful migration projects, there has been some pain points in this migration and modernization field in the past decade. The bottlenecks that prevented large-scale migration projects in the early attempts within 1990ies can be listed as follows:

- Lack of equivalent target platforms (hardware and application/transaction servers) with comparable performance with similar size.

- Lack of complete or extensive technical or management level methodologies to deliver these projects.

- Insufficient migration aids or tools to automate the migration processes (as described above)

- Insufficient level of research in the field of migration and related fields. Many of the initial researches in the field not suitable to be applied in the real life projects and the concentration was mostly on the proof of concept level not targeting further application of the proposed ideas and prototype tools to be used in large scale real life projects.

On the other hand, some of the research studies in the field were successful but the industrialization of the outcomes of these research studies together with the application of them in projects took considerable amount of time.

Currently, in all of these areas, there have been many advances, which enable large-scale migration projects. Both on academic side and industry there are and there will be continuous efforts to contribute in this migration area in order to cope with the advances in the target platforms and architectures (OO technologies, SOA, Grid computing, Enterprise Application Integration Technologies, Virtualization etc)

2.2.3. State Of The Art

Based on the above listed problem areas, the current status in the migration field is described here under three main categories: current research topics, advances in the migration tools and vendors in the industry and advances in the target system environment. Pertaining to the thesis study topic, the first two items will be discussed in more detail in the following sections.

2.2.3.1. Migration Related Research

The early migration related research was targeting the migration from these monolithic and procedural mainframe systems toward distributed architectures like client-server architectures [26] [27] [28] [29]. Another research area is the migration towards distributed object architectures and technology. Several efforts

are carried out in the field aiming to migrate to these target architectures [3] [30] [31] [32].

On the other hand, in parallel to the advances about web services, Service Oriented Architectures (SOA) and Service Oriented Computing, many researchers have concentrated in this area as well on the most optimal way to migrate the mainframes in relation with this technologies to the target platforms [18] [33] [34] [35]. Erradi [18] states that one of the key obstacles for SOA is the presence of legacy applications that support critical business processes but are inflexible and hard to adapt for integration with other business applications. To guide the architects in selecting the optimal combination of legacy modernization options, the need for a tool based decision framework for modernization options is emphasized [18].

Another contribution is provided by [36] with the proposal of a technique called Service Migration and Reuse Techniques, so called SMART. SMART process aims to support organizations to make initial decisions about the feasibility of reusing legacy components as services within an SOA environment. In order to do that, specific interactions that will be required by the target SOA environment and any changes that must be made to the legacy components are identified. To achieve this, "SMART gathers information about legacy components, the target SOA environment, and candidate services to produce (1) a preliminary analysis of the viability of migrating legacy components to services, (2) an analysis of the migration strategies available, and (3) preliminary estimates of the costs and risks involved in the migration." [36]

For the migration to Web services and service-oriented architectures, the most critical issues are considered as program comprehension and analysis, legacy system migration issues and the technology and standards issues [37]. Regarding the legacy system migration and evolution issues, the following are the active research areas of focus [37]:

1) Migration and evolution processes towards network-centric platforms with emphasis on evaluating migration techniques, assessing target platforms, and planning the migration effort.

2) Migration and evolution techniques with emphasis on system and source code transformations, model driven approaches as well as, automatic or semiautomatic generation of wrappers and mediators.

3) Control and data integration issues, with emphasis on the use of legacy assets in workflows, issues in transaction management, logging, and diagnostics.

4) Quality issues for the migrant system with emphasis on requirements modelling, association of requirement models with design and transformation decisions as well as, assessment of alternative target designs with respect to the requirement and quality objectives for the migrant system.

The modelling of legacy code and user interface reengineering are also specifics areas of research in this field [38] [39] [40] [41] [42]. Jianju et al. discusses that [42] in order to make a better use of these legacy COBOL systems, research on modelling and reusing COBOL code are significant and presents the following approach with UML Collaboration diagrams:

"Firstly, legacy COBOL code is transferred into WSL, which is an intermediate language. Secondly, WSL representation is restructured to eliminate GOTO statements. Then an object concept model is created by operating WSL representation. Finally, the collaboration diagrams are constructed based on this object concept model. These collaboration diagrams express the dynamic behaviour of the system, using the structural class and relationship elements of the model, and facilitate engineers to understand and reuse legacy COBOL code [42]."

In relation to the migration management methodologies as proposed within this thesis study, there are also several studies [3] [12]. [3] handles specific case with the migration to object oriented technologies from a reengineering and reverse engineering perspective and offers a migration process and a "Migration Project Support System" model to conduct the migration as a reengineering practice. [3] His work mostly concentrates on the specific techniques to be used during the migration projects certain stages like static featuring technique, rule-based class recovery techniques for object identification.

Lucia et al. proposed an approach using a wrapping technique to migrate legacy systems written in RPG into object-oriented platform [43] [44]:

"It contains six steps to gradually migrate the legacy code:

1. *Static analysis of legacy code.*
2. *Decomposing non-batch programs.*
3. *Abstracting an object-oriented model.*
4. *Re-engineering the system according to the results of decomposition and abstraction.*
5. *Encapsulating the identified objects into wrappers.*
6. *Incremental transition of object wrappers."*

Although this approach is advantageous in maximizing the reuse of legacy code by means of the incremental wrapping approach, on the other hand it is difficult to decompose the applications in the legacy system, which is further increasing the complexity of the migration.

Similarly, several methodology proposals are available from different industry vendors [45] [46]. But the methodologies offered are heavily based on their related migration products and technologies. On the other hand, in an analysis report conducted by Ovum [9] it indicated that there is still no established 'best practice' for renewal: "The IT services industry is stepping up to offer legacy renewal practices in response to a groundswell of need from their clients. Most service providers also offer an initial 'Portfolio Assessment' phase to assess the current situation and to recommend a roadmap to achieve renewal of large complex installations in several phases using a range of approaches. These companies can reduce your risk, but there are no guaranteed outcomes, so audit their processes carefully".

The difference of our proposed methodology framework is that it foresees the usage of migration tools within the process but not dependent on any specific vendor tool or vendor and another main difference lies in the definition of all stages during the life cycle, i.e., clear description of all input and outputs of each stage is given within the proposed framework.

2.2.3.2. Major Migration Tools And Vendors In This Field

The reason why we devote a specific section to the migration tools is that in a real life migration projects, these types of commercialized solutions are needed especially for medium to large migration projects. From the other perspective, it would be a misperception to assume that the whole migration is completely taken care completely by these tools and everything; with a 100%, automated process is migrated to the target platform. In a recent study on language conversion, Terkhov explains this point as "Language conversion is a laborious process. Achieving the maximum efficiency of conversion without compromising the quality of converted system is the programmers' dream." [47].

In an analysis, report by Ovum [9], it is stated, "There is no 'one size fits all' solution. The first generation of non-invasive legacy renewal tools are mature and provide a sound basis for legacy re-use. A second generation of tools are emerging to provide more comprehensive renewal choices. There are choices to support a broad range of business needs, from simple web access to customer information held in legacy systems, to a full overhaul to align IT with new business imperatives such as agility, globalisation, compliance and straight-through processing (STP)."

The correct positioning of these tools is regarded as tools to assist in migration projects and to automate the migration process as much as possible. This is needed to decrease the project durations and effort by this automation and nevertheless eliminate or minimize error-prone manual intervention or via the usage of these tools.

The migration approaches and the corresponding migration tools can be classified as follows:

- **RE-HOSTING**: Emulation tools, COBOL compilers in target environment, CICS transaction environment simulators in Windows, OO-COBOL integrated development environments, Batch/JCL processing tools.
- **LANGUAGE CONVERSION:** Conversion Tools like from COBOL to JAVA.

- **REENGINEERING:** Legacy Analysis Tools, Business Logic Extraction Tools.

Below, a more detailed description of each main migration approach is provided with the supporting tool references from different vendors. More detailed analysis of these approaches and tools is beyond the scope of this study.

- **RE-HOSTING**

Re-hosting is migration of mainframe applications (in our case COBOL programs) from the mainframe to target platform (typical Windows or Linux) with minimal change in the application code. In this scenario, it is not enough only to migrate the COBOL applications but also the CICS environment and Batch environment has to be considered as well. In order to enable CICS environment replication CICS-compatible transaction servers are used or CICS related part of applications are reengineered to work under standard transaction servers available in the industry. Microfocus [45] and Fujitsu [48] are two leading re-hosting vendors. These solutions are also referred as "software clones" by [49].

Micro Focus Server™ for Mainframe Migration provides a high performance, scalable deployment environment for hosting applications that have been migrated from IBM mainframe systems. It also provides the infrastructure to support the integration of migrated applications with technologies such as .NET, J2EE, or SOA to satisfy evolving business requirements. The Micro Focus Server product is built around a transaction technology framework and Job Execution System (JES) engine that supports IBM JCL™ job streams, IBM CICS™ transactions, IBM IMS™ transactions, Web services, and COBOL/J2EE integration within a composite environment.

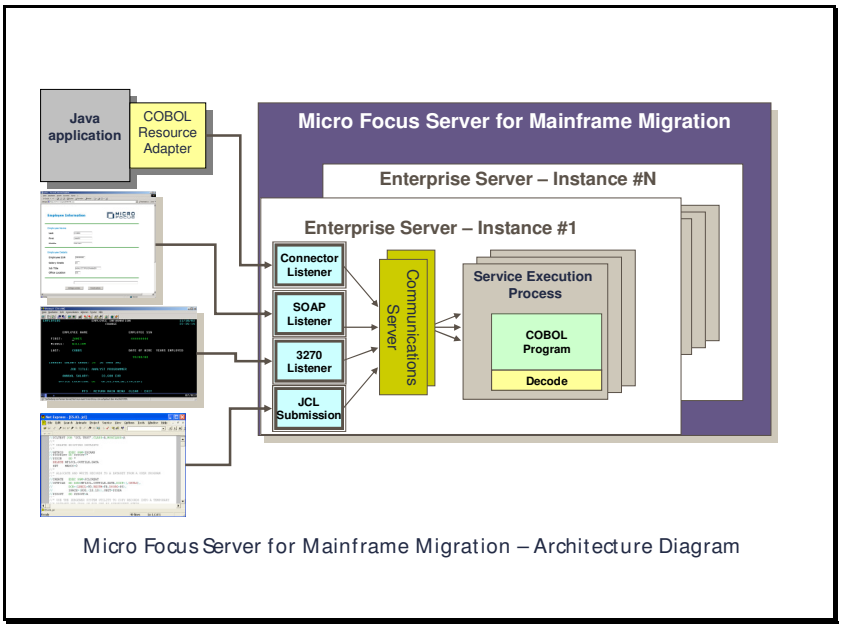

Figure 1. Micro Focus Server Architecture

Complementing this Execution environment, a development and test environment solution to manage the migrated COBOL / CICS programs is also provided by the vendor.

Fujitsu Software [46] is also providing similar a similar solution called NETCOBOL, which executes in a Microsoft.NET environment. It is added to Visual Studio.NET developer suite as an equal development language to Visual Basic or C# [49]. Within the solution package the CICS emulator called NeoKicks, and as well as a batch JCL runtime product called NeoBatch is provided.

Additionally, there are several open source projects such as COBOL for GCC, TinyCOBOL, and OpenCOBOL. These COBOL compilers are freely available on both mainframe and distributed platforms.

In the rehosting approach, the major advantage is that the application code is completely reused with no or minimal changes and therefore compared to other approaches the migration takes shorter time and less effort. Once they are migrated to the new platform, it becomes possible to use the advantages of the new development environments, architectures and tools offered by the new target

platform both in technical and financial aspects. On the other hand, as the code is migrated with no changes or improvements, there is neither added benefit of reengineering nor change of the language. Although, the COBOL is executed in more modern and effective development environments like Visual Studio, it is still COBOL code. This option is suitable for organizations that have enough COBOL resources and looking for cost effective execution platforms and a fast migration approach. Another critical consideration is whether the existing applications are serving the needs of the organization in terms of business processes. If this is not the case, than considering a rehosting approach would not be the first choice of migration approach.

- **LANGUAGE CONVERSION**

As the name implies, this approach covers conversion of the mainframe legacy language to the desired language in an automated process. Achieving the maximum efficiency of conversion without compromising the quality of converted system is the programmers' dream [47]. This is an area where many efforts were spend by both academy and industry.

The language conversion is provided either as a service or as the conversion product. Bluephoenix [45], ATERAS [50] and Metaware [51] are some of the vendors in the industry.

The conversion from 4GL or Proprietary COBOL to industry standard COBOL is successfully managed by many tools [53]. In addition, in many cases, just being able to convert to industry standard COBOL that can be executed in modern target platforms would be sufficient for many organizations. On the other hand, conversion from 3GL COBOL to OO languages like C# or Java is still not much satisfactory in terms of code readability and comprehension. Although these conversions are fielded and tried in many real life projects also, yet the conversion tools are not at the ideal maturity from this aspect readability.

However, on the other hand, with the language conversion approach, compared to re-hosting solution, greater opportunities to modernize the source code, databases and user interfaces are presented to the organizations as all the code will be in JAVA or C# for example. Another suitable case can be the organizations with a code base that requires very little change or improvement at business process

level that does not concern a lot about the code readability after conversion but more interested in the shut down of the mainframe for decreasing mainframe maintenance costs.

- **REENGINEERING**

The reengineering approach includes a broad range of alternatives. However, discussion of each of them is beyond the scope of the thesis. Common to all the reengineering approaches is the need for a legacy analysis tool. These tools can also be used in other alternatives like re-hosting or complete rewrite. Currently available legacy analysis tools provide a portfolio analysis and extract the relationships and dependencies among all application elements. Within the proposed migration management framework, it is assumed that these tools should be used during the initial analysis and planning phases.

These tools provide core data for application portfolio assessment, and additionally application analysis and business rule discovery. Typically all the data structures, data flows, process diagrams can be identified via the tools. With some of these tools, it is also possible to do some heuristic analysis of source code patterns. These tools can be used to not only for migration but also for everyday maintenance, component mining or rewriting of legacy applications. They support the decision making process by providing understanding of the legacy system. Some of the tools have the capability to do lexical, syntactical, and semantic analyses of a system's components to determine its structure and the relationships between and within program entities.

CHAPTER 3

MODERNIZATION STRATEGY SELECTION ALTERNATIVES

3.1. Comparative Analysis Of Modernization Strategies

Before the selection of the migration approach, it is vital that a strategy selection analysis needs to be performed between four main alternatives. To recall; these four main categories of "modernization strategy" were defined as follows in Chapter 2:

1. Do Nothing: Although from technical perspective there are many technological alternatives for migration, with non-technical considerations (managerial, financial etc) considerations, "Do nothing" is diminishing but still viable option based on the organization's status.

2. Packaged Application: Commercial of the shelf (COTS) application option involves moving completely to a standard package in the new environment like the standard ERP applications.

3. Rewrite: This option is mainly based on the complete or partial rewrite of the applications. This option can be handled either by re-using the business logic available within the existing applications. Therefore within this option it is critical on how the business logic extraction from these programs is handled.

4. Migrate: Among all the strategies, the decision to migrate is a checkpoint for further categorization. Once this alternative is chosen, the further categorization of the migration approach like language conversion, re-hosting, re-engineering, wrapping etc is done based on the existing and target environment and the underlying technological components and migration tools.

Following is a high-level comparison of these approaches from risk/cost, percentage of reuse and added agility/benefit perspectives:

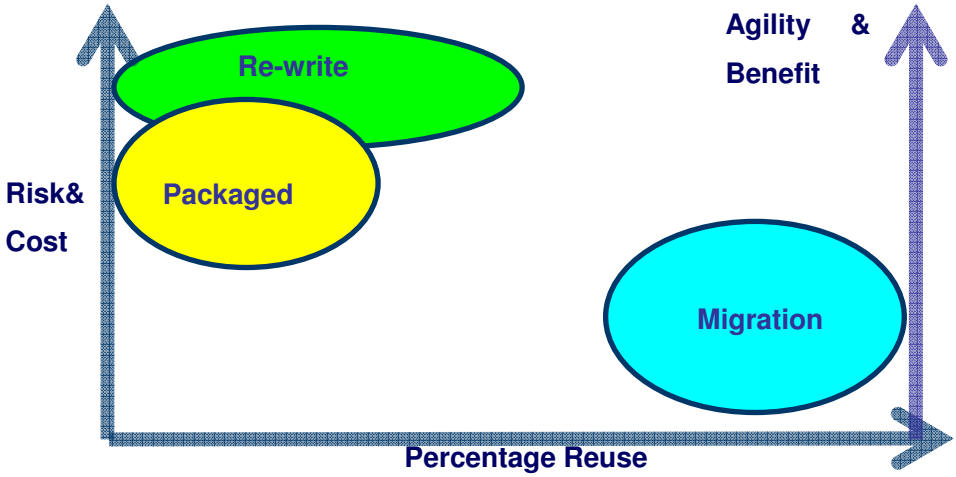

Figure 2. High Level Comparison of Modernization Alternatives

It is clear that among all alternatives re-write is the highest risk option but the benefit for the organization , on the other hand is also very high as everything is re-developed in the target platform with the new programming languages and technological components, middleware, databases etc. The cost of this approach is therefore higher than the other alternatives. The percentage of re-use for rewrite strategy is typically not the actual code level but on business and program level logic level, which can be extracted from the existing applications with different methods or tools like business logic extraction or analysis tools.

On the other hand, for packaged option, the cost and risk factor is mostly lower than rewrite but much higher than migration option and the level of re-use is much lower as the processes will be almost totally discarded and the packaged application will be customized to a certain extent if needed. Therefore, this option is mostly for organizations with commodity type of business needs and looking for low cost execution platforms to mainframes where the current application is not supporting the current business needs. However, from the risk perspective, unlike the general perception the risk and cost of the modernization is higher than migration.

For migration, with the advances in research in the field together with the emergence of improved migration tools to assist migration projects, the migration among others is the least risky and less costly approach. However, as the application code re-use is very high, which means; mostly the code remains in COBOL, and some software clones are needed for rehosting in the target environment, these can limit the added benefits or agility compared to especially re-write alternative. There can be new dependencies due to these softwareclones and OS/CICS emulators. For migration, it is also critical whether there is enough IT staff to maintain and improve the applications in the new platforms with the migrated programming language.

3.1.1. "Do-Nothing" Modernization Strategy

As the mainframes were initially designed with monolithic architectures and integration with other systems or thru web were not even in the list of initial design parameters, mainframes were expected to work by them selves. Therefore, the need for integration with external systems or networks is one of the reasons why organizations are having problems. Because the integration is a fundamental and inevitable business need for nearly all organizations. To overcome this problem, some short-term solutions were developed within the industry throughout the years. However, these solutions are not solving the fundamental problems with the mainframes but only delaying the migration for a while.

Another difficulty with mainframes is the cost factor. For many organizations, maintenance costs of a mainframe compared to a typical Windows

based server environment is tremendous. This is also another reason why this alternative is getting less and less attractive.

The problems with the mainframe and the motivation for the organizations to migrate off these systems were also explained in detail in previous sections within the thesis.

A recent Gartner report [53] is commenting on why this "do-nothing" option is getting more difficult, as follows:

"Alternatively, organizations have run a series of tactical projects to shore up existing vertical technologies within the infrastructure, without having the charter (or funding) to address the overall problem. This task-oriented view is, in many cases, forcing companies to implement point solutions to solve problems that were caused by the last tactical project (for example, consolidation leads to virtualization, which leads to server sprawl, which, in turn, could lead to power and cooling issues sooner than expected). IT leaders are looking to modernization as a systematic approach to improving the overall infrastructure in a logical, stepwise approach, rather than as a continuous fire fighting exercise. This is not about implementing new technologies or redesigning the infrastructure, but it is about bringing strategic planning back into the equation for IT operations. Nothing gets done unless it fits into the strategic plan — and unless the consequences of that action (that is, the expected cascade effects) are understood and the effects are integrated into the strategy."

Metagroup survey [54] lists the top business driver prompting modernization of legacy systems as "Agility and Adaptability".

When compared with other modernization strategies, there is no risk of migration as there is no migration decision. On the other hand, there is no cost savings or added agility or benefits gained with this strategy.

3.1.2. "Packaged" Modernization Strategy

These packages are typically enterprise resource planning (ERP), customer relationship management (CRM) or supply chain management (SCM) packages or solutions. For the organization with standard business needs that can be addressed by these tools, the packaged applications can be the most appropriate solution.

However, if the organizations business needs are very different than what these packages can offer, then the result can be a catastrophe, where there is a new solution, which is not answering the needs of the organization, and therefore the customization is taking longer than expected and in the mean time the operation is continuing in the existing mainframe. Therefore, this is a scenario that results in the maintenance of both systems for an unexpected long transition period.

Following are some major benefits of this strategy:

- Potential cost advantages gained by moving off the costly mainframe environment (depending on the chosen packages: the costs advantages gained and the cost of the package needs to be analyzed for the strategy selection)
- IT resources availability (i.e. COBOL / CICS programmers) is no longer a concern as the core application is managed by the vendor where the organization becomes a user of the packaged application.
- The vendor carries the responsibility to improve the product based on the new technological advances.
- Usage of state-of the art product and technology enabling increased agility or integration capabilities (based on the chosen packages)

Following are major disadvantages of this strategy:

- High modernization costs (typically underestimated)
- High modernization project risk (especially if there is high business differential from the standard packages)
- Danger in loosing advantage in competition due to giving up to own applications and processes (loss of business differential)
- Costly modifications to make the package fit the business – or worse – to make the business fit the package [55].
- Retraining costs for the users

3.1.3. “Re-Write” Modernization Strategy

The rewriting alternative is mostly suitable in cases where the applications are no longer matching the current needs of the business and the processes of the organization cannot be handled by the standard packaged solutions. On the other hand, as there is no dependency on the target platform, programming language, development tool and environment selections, this gives freedom to the organization to upgrade and improve the business processes as well via this project as a second major gain: both the mainframe related problems are solved with the turning off the mainframe and also the business processes can be redesigned and improved.

However, all these benefits come with a high cost and risk. Unfortunately many rewrite projects have either completely failed or completed with much higher costs and project durations than planned. The enterprise level mainframes are typically consisting of several millions line of code and many supporting utility programs, it is a high-risk challenge to rewrite all these applications at once. A safer and more feasible approach would be to rewrite the applications incrementally. The applications to start first can be selected with the help of an application portfolio analysis and business priorities evaluation study.

Following are some major benefits of this strategy:

- Cost saving achieved via shut down of mainframe
- Agility and flexibility gained with the newly developed application
- Freedom in the selection of the target platform.
- Possibility to do a business process redesign and optimization
- No dependency on the mainframe related competencies.

Following are major disadvantages of this strategy:

- Very high cost of modernization project
- Long deliver time for the project

- Very high risk of failure, cost overruns and delays (esp. with the rewrite of whole system)
- High transition period and its side effects due to long project implementation periods
- Retraining costs for users and IT staff.

3.1.4. "Migration" Modernization Strategy

As described in previous chapter, migration itself contains several approaches within itself. This approach is especially important for organizations, which have applications addressing the needs of the business, but is under severe cost pressure due to mainframe operations and mainframes. Because migration by default foresees the reuse of existing applications in the new target environment either in a rehosted form or through language conversion. Both cases assume the usage of existing business processes as before. This brings another advantage in minimization of training needs after migration.

On the other hand, this approach is providing less agility compared to the rewrite approach as there is still dependency on the software clones or emulators in the new environment. However, the migration tool vendors are trying to tolerate this drawback by native integration to the state of the art development environments like Visual Studio .NET. The obvious advantage comes with the minimal risk and costs associated with the migration project itself.

Following are some major benefits of this strategy:

- Cost saving achieved via shut down of mainframe
- Low migration periods and costs.
- Agility and flexibility gained with the newly target platform compared to the mainframe environment.
- Maximum re-use of existing assets and IT skills
- Low impact on business operations

Following are major disadvantages of this strategy:

- Relatively lower agility compared to rewrite.
- Some dependency on the migration tool and platform vendors.

3.2. Modernization Strategy Selection Method

The selection of migration strategy can be regarded as a strategic decision, whereas the selection of specific migration approach can be regarded as a tactical decision. Based on the above findings about the modernization strategy alternatives, below given method is proposed for the selection of the appropriate strategy for each organization.

Figure 3 and Figure 4 summarize the high-level rationale of the selection method below:

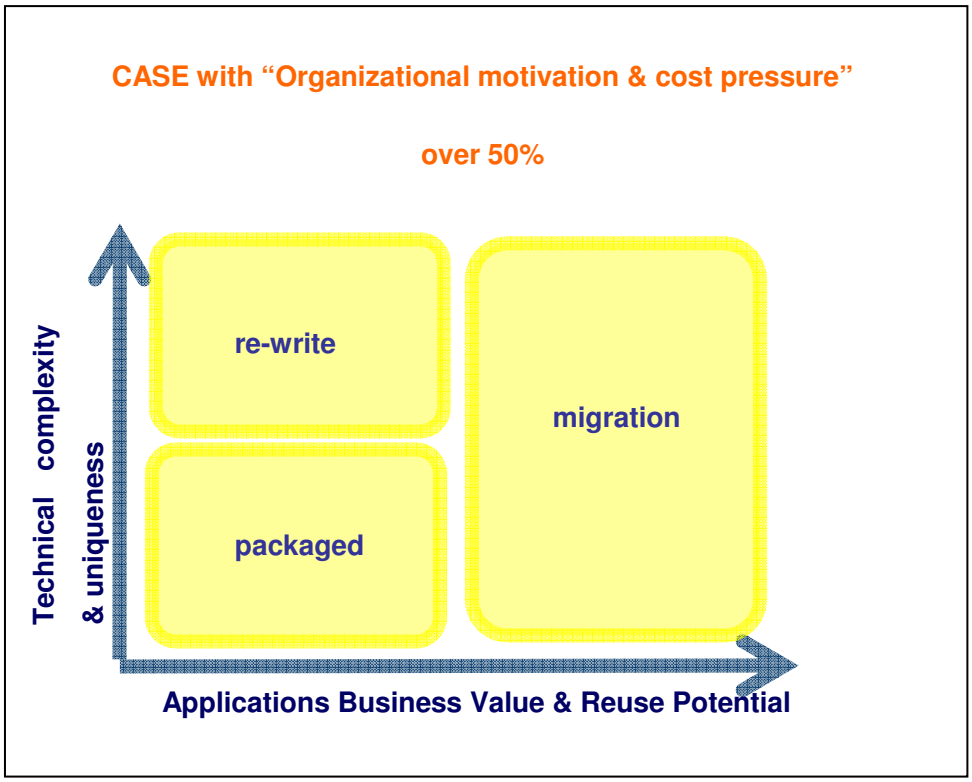

Figure 3. High Level Comparison of Modernization Alternatives Motivation & Cost Pressure" over 50%.

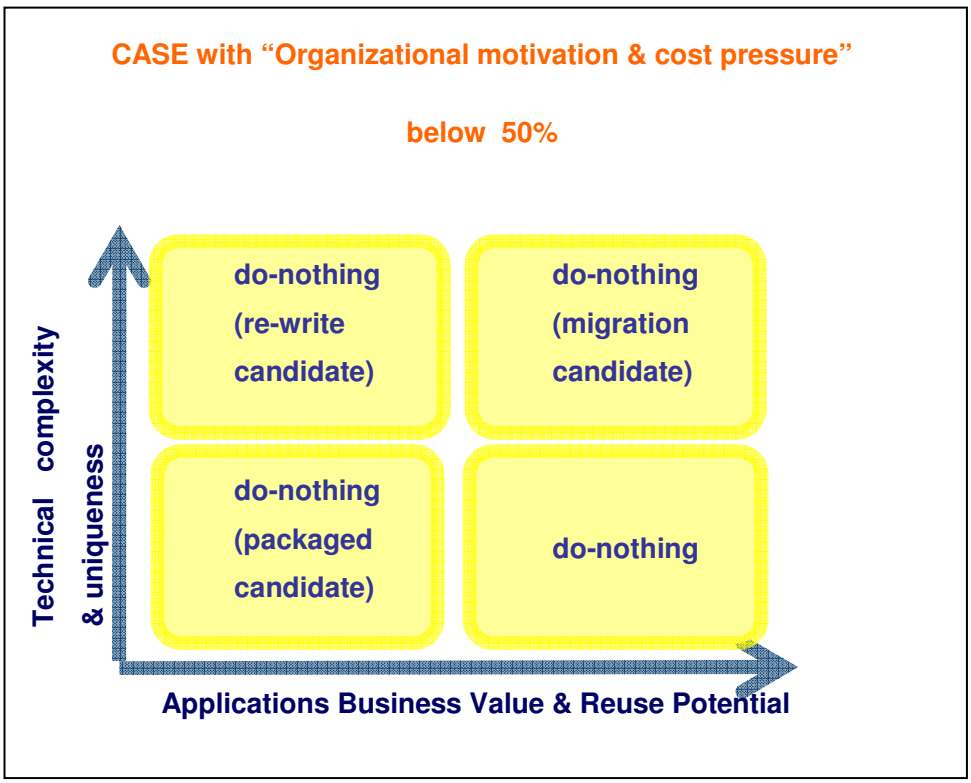

Figure 4. High-level rationale of selection method with "Organizational Motivation & Cost Pressure" below 50%.

As depicted in the figure, the factors that shall be considered for the selection of modernization strategy can be classified under three main dimensions. Although looking from different perspectives, this categorization may vary based on the opinion of the researchers; the categorization given here is based on the professional experience with various real life projects and the research in this field.

The first dimension covers the technical complexity and uniqueness related parameters. This dimension is for the assessment of the current environments complexity and the uniqueness of the business applications of the organization.

Table 2 Criteria for Technical Complexity & Uniqueness

Evaluation Factors	**Total Score**	**Evaluation Method**	
Architectural Complexity (CICS, JCL, Other Host Facilities & Utilities, File structures)	20	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Business processes with business differential	35	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Lines of Code	10	< 100 KLOC	10%
		< 500 KLOC	40%
		< 1 MLOC	70%
		> 1 MLOC	100%
Multiple deployment and development environments and programming languages	10	Single	10%
		2	40%
		3	70%
		> 3	100%
MIPS Grade	15	< 1000	(MIPS /10)%
		> 1000	100%
Code complexity	10	Low	20%
		Medium	50%
		High	90%

The second dimension is for assessment of business value and re-use potential. The two areas are evaluated in the same dimension because the two factors are valuable only when both of them are there at the same time. If the applications are satisfying the needs of the organization but not possible to re-use than it has very little value for the selection of the strategy.

Table 3 Criteria for business value and re-use potential.

Evaluation Factors	**Total Score**	**Evaluation Method**	
Satisfaction of current business processes	40	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Adaptability& flexibility of processes for future changes	15	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Availability of good coding standards	15	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Availability of documentation (programmers/system users manuals, system documentation, design docs. etc)	10	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%

Table 3 (continued)

Availability of experts (IT staff, domain experts)	20	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%

Final dimension is devoted to Organizational motivation and cost pressure factors. This factor is nevertheless the most important one for the success of the project. The willingness of the organization for modernization can be low due to varying technical or other reasons (job safety concerns, resistance to change, technical complexity related issues, concerns about negatively impacting current business operations, pressure from current environment vendors, negative lobbying from current system support service and product vendors, etc). In cases where this motivation is low, the choice of re-write might not be the best selection due to possible misalignment within the organization, or the factors for this shall be carefully analyzed and studied in order to increase the buy-in within the organization. So the case with this dimension can be evaluated under two scenarios: “over 50%” and “below 50%. If there is no cost pressure and the organizational motivation is low for modernization, this is typically the case for “do-nothing” strategy in short term. However, depending on other factors, in long term the strategy can be to look for other modernization alternatives as depicted in Figure 3.

Table 4 Criteria for organizational motivation and cost pressure

Evaluation Factors	**Total Score**	**Evaluation Method**	
Cost of Ownership compared to modern target platforms (perceived or factual)	25	Lower	10%
		Same	30%
		Higher	80%
		Much Higher	100%
Availability of organizational resources for modernization	15	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Performance/scalability issues with the system	15	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Issues with maintenance	10	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%
Issues with the new features implementation	10	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%

Table 4 (continued)

Issues with IT budget.	25	Very low	20%
		Low	40%
		Medium	60%
		High	80%
		Very High	100%

3.3. Different Organizations

Three different organizations were selected for this purpose. The organizations' profiles were studied through questionnaire and one-to-one interviews.

The first one is a private bank with a set of banking and credit campaign management applications hosted in the mainframe, which was looking for a modernization solution to close down the mainframe and move to environments that are more modern with more flexibility and agility and at the same time with lower cost of ownership.

The second organization is a public institution with a very large mainframe system. They were also searching for the best modernization strategy for their organization. They had a very complex system with complex business applications and architecture. Their code size and code quality was very high. The overall organization's overall IT strategy was towards open systems and therefore they were concerned about being consistent with that strategy in the selection of the modernization strategy. Due to the size of the organization and number of stakeholders in the environment, there was a mixture of opinions in favour and against any modernization option.

The third organization is a private hospital looking for a modernization solution from their medium sized (MIPS) mainframe. "Do-nothing" is was not an option for them as there were severe complaints about the existing legacy hospital

management solution on both performance and functionality side. Their architecture was not complicated but the application size was over 1 million lines of code (MLOC). The highest priority for them is not to harm the ongoing operations in the hospital and to make sure the transition from mainframe to new system is done smoothly.

Below tables, list the results of the application of the method to these three organizations based on the given criteria:

Table 5 "Technical Complexity & Uniqueness" Results

Evaluation Factors	**Total score**	**Bank**	**Gov't Inst.**	**Hospital**	**Bank**	**Gov't Inst.**	**Hospital**
Architectural Complexity (CICS, JCL, Other Host Facilities & Utilities, File structures)	20	Med.	High	Low	4	16	12
Business processes with business differential	35	High	Very High	Very Low	28	35	7
Lines of Code	10	< 500 MLOC	> 1 MLOC	< 1 MLOC	4	10	7
Multiple deployment and development environments and programming languages	10	2	2	single	4	4	1
MIPS Grade	15	600	> 1000	250	9	15	3,7
Code complexity	10	High	High	Med.	10	10	5
				Total	**59**	**90**	**35,7**

Table 6 "Business Value & Re-use Potential" Results

Evaluation Factors	Total score	Bank	Gov't Inst.	Hospital	Bank	Gov't Inst.	Hospital
Satisfaction of current business processes	40	Low	Very High	Very Low	16	40	8
Adaptability& flexibility of processes for future changes	15	Very Low	High	Low	3	12	6
Availability of good coding standards	15	Med.	Very High	Med.	9	15	9
Availability of documentation (programmers/system users manuals, system documentation, design docs. etc)	10	Med.	High	Low	6	8	4
Availability of experts (IT staff, domain experts)	20	Low	Med.	Low	8	12	8
				Total	**42**	**87**	**35**

Table 7 "Organizational Motivation and Cost Pressure" Results

Evaluation Factors	Total score	Bank	Gov't Inst.	Hospital	Bank	Gov't Inst.	Hospital
Cost of Ownership compared to modern target platforms (perceived or factual)	25	Higher	Much Higher	Higher	20	25	20
Availability of organizational resources for modernization	15	Med.	High	High	9	12	12
Performance/scalability issues with the system	15	High	High	High	12	12	12
Issues with maintenance	10	Very High	Med.	High	10	6	8
Issues with the new features implementation	10	High	High	Med.	8	8	6
Issues with IT budget.	25	Med.	High	High	15	20	20
				Total	**74**	**83**	**78**

Below is the summary table and analysis of the results:

Table 8 Summary of Results & Recommended Strategy

	Technical Complexity & Uniqueness	Business Value & Re-use Potential	Organizational Motivation & Cost Pressure	**Recommended Strategy Based On The Selection Method**
Bank	59	42	74	**RE-WRITE**
Gov't Inst.	90	87	83	**MIGRATE**
Hospital	35,7	35	78	**PACKAGED IMPLEMENTATION**

Positioning within the conceptual graph shown below is also graphically displaying the selection of the recommended modernization strategy. The displayed results in turn validate our selection method also in graphical representation.

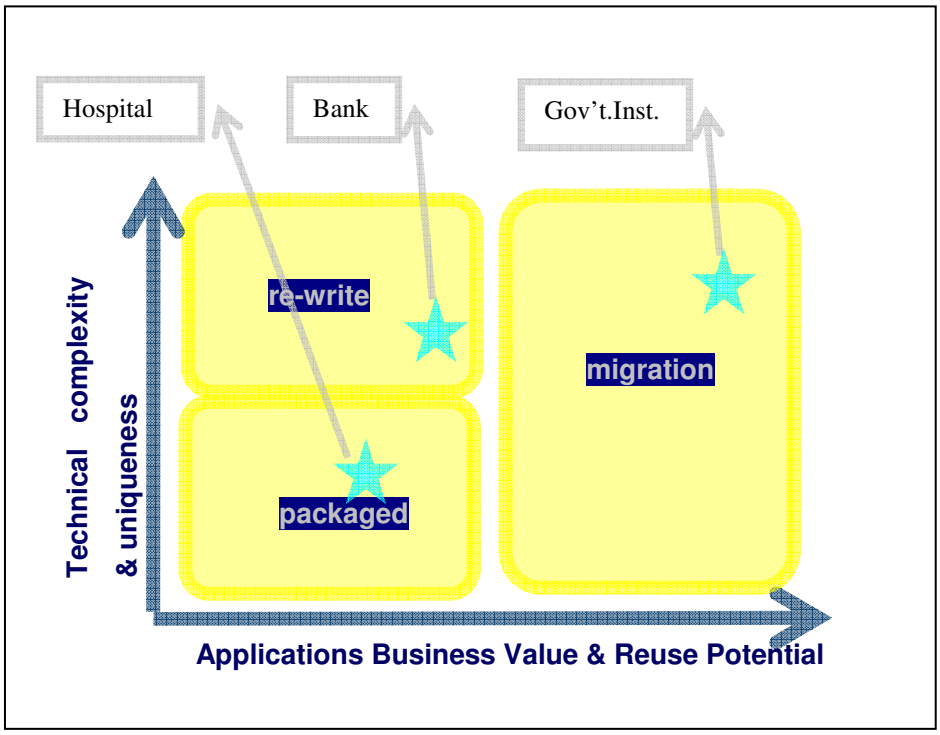

Figure 5. Positioning of the strategy selection

Without these type of indicative studies, IT managers would tend to make development decisions based largely on instinct. But in highly complex IT environments, this needs to be done based on solid data with systematic decision processes. Therefore the proposed decision framework provides a set of criteria with measurable and quantitative metrics. Another important topic for this decision methods is that it is not biased in favor of any specific solution or approach proposed by vendors in the industry. After the application of the method, the specific recommended modernization strategy needs to be worked in detail for further analysis and positioning in order to determine the right implementation approach within the selected strategy.

CHAPTER 4

MIGRATION MANAGEMENT FRAMEWORK COMPONENTS

4.1. Introduction

As described in previous chapters, mainframes typically form the backbone of information flow within a legacy organization and are normally mission-critical; a failure in one of these systems may have a serious business impact. They contain business rules and workflow controls that help organizations process orders, track revenues, manage inventories and perform other key business functions. On the other hand, to recall, the problems these massively complex systems pose include:

- They cannot evolve to provide new functionality required for their host organization to remain competitive;
- They run on obsolete hardware which is expensive to maintain and may reduces productivity due to its low speed;
- Maintenance is expensive, tracing failures is costly and time consuming due to a lack of documentation and a general lack of understanding of the internal workings of the systems;
- Integration efforts are greatly hampered by the absence of clean interfaces.

Switching off a thirty year old system and plugging in a new feature rich replacement overnight is not an option. Attempting to recover the business logic and constructing replacements is also too risky and costly.

Legacy system migration is concerned with developing a target system, which retains the functionality and data of the original legacy system but which can be easily maintained and adapted to meet future business requirements. Unfortunately, the replacement of legacy information systems is a far from straightforward process; such a project could, quite legitimately, address the areas of reverse engineering, business reengineering, schema mapping and translation, data transformation, application development, human computer-interaction and testing.

To address this problem, Migration Management Framework offers a thorough methodology allowing the migration of legacy systems to new platforms and architectures. The methodology is, where applicable, supported by a set of proposed migration tools and accompanying controllable processes. With an IT project management view and systematic, controllable migration processes combined with set of migration tools or methods is vital to the success of these migration projects.

4.2. Process View

Migration Management Framework is established as a refined baseline with the aid of best practices available in the migration and software development specialty areas. Architecture of the framework is process based, repeatable and open for improvement.

The framework covers a lifecycle proposal with the processes, activities, tasks, work products, tools and techniques. A list of available tools and products from the migration tool vendors are also referenced within the framework as potential alternative tools and methods that can be applied at certain phases of the lifecycle.

Framework as presented in the following figure, provides primary migration processes as Legacy System Analysis, Design (Migration Implementation Planning), Implementation and Deployment.

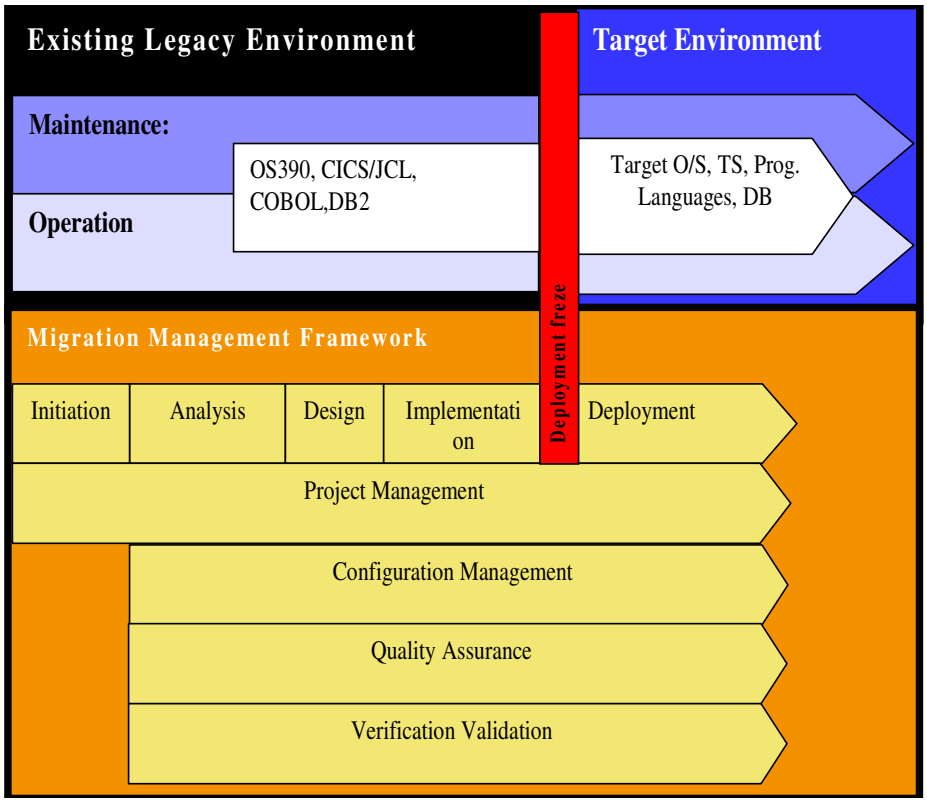

Figure 6. Migration Management Framework High Level View

- The purpose of Legacy System Analysis is to obtain an in-depth understanding of the legacy system to determine an overall roadmap for migration efforts and to document migration requirements specification after identifying and establishing the legacy baseline. A Migration Implementation Strategy is determined in accordance with the complexity of the legacy environment and risk associated with the migration. This phase is crucial for the success of the migration

effort. Output of Legacy System Analysis is documented in Migration Requirements Specification.

- The purpose of Design (Migration Implementation Planning) is to design and document in detail how the legacy system will be migrated to the target environment based on the Migration Requirements Specification and to provide a technical plan that will provide a guideline to determine the progress of implementation. During the design, the project team (including the members from the organization) will establish a logical solution - target architecture diagrams and database design description documents. The host organization or company or institution that owns and/or operates the mainframes and which is also responsible for the migration project is referred as “Organization” throughout the lifecycle. Next step of design phase is to determine a roll-out strategy for the organization by dividing the migration into many steps. Each rollout focuses on migration of a certain part of the target system. Migration Implementation Plan is a technical document describing how the legacy migration will be performed. Further analysis and design related to rollouts (increments) will be detailed during the implementation phase. Legacy system analysis and design efforts are relatively more challenging than other phases as it covers consequential activities affecting the entire migration effort. At this point, the solution is "logical" because it exists on paper or in a design tool. This logical solution described in detail in the Migration Implementation Plan is then passed to the Implementation Phase, where it is turned into a physical solution.

- Implementation Phase consists of Development Environment Preparation, Increment Based Analysis and Design, Legacy System/Data Conversion, Integration and Acceptance. The purpose of the Increment Based Analysis and Design is to obtain detailed information on legacy system components to be converted during the increment and to obtain verification references related to the target components. Legacy Conversion, correction and verification is performed incrementally in development environment. Converted

and verified components are transferred into test environment for integration and validation. Integration is a complex, ongoing process; often integration and testing of one component requires involvement of other components and legacy system. Due to mission critical nature of the legacy information systems, it is imperative that there shall be no inconsistencies between the output of a legacy system and that of its replacement system. The implementation phase is also where the project team creates the support documents and training content related to the target platform. Validated solution and documentation are submitted for acceptance. The purpose of acceptance testing is to obtain formal end-user approval in accordance with the project plans, prior to the deployment of the solution.

- The Deployment Phase is concerned with the cutover from the legacy system to the target system. When dealing with legacy systems, this process must cause as little disruption to the business as possible. The deployment phase results in the orderly transition of the enterprise from dependence on the existing legacy system to usage of the newly migrated system. This is not necessarily an instantaneous transfer, but it is a staged rollout of the new application, a gradual replacement of the current legacy system functions. Below figure presents the lifecycle model for the analysis, design, implementation and deployment phases that was briefly described above.

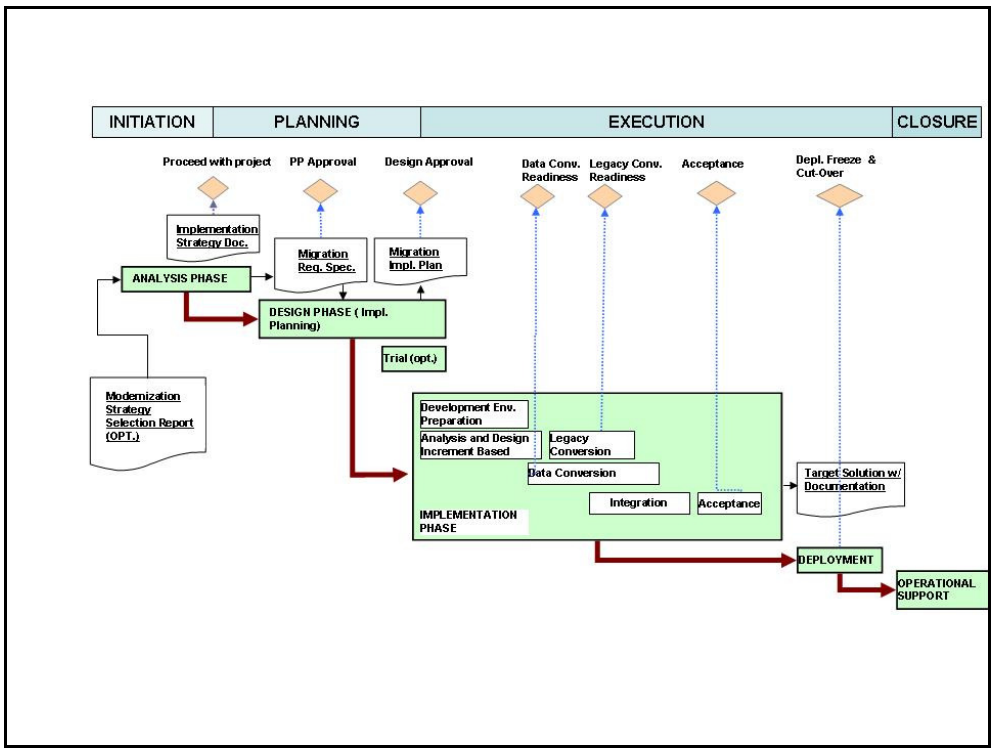

Figure 7. Lifecycle Model View

CHAPTER 5

MIGRATION MANAGEMENT FRAMEWORK

5.1. Guiding Principles And The Underlying Rationale

Migration is a complex process involving significant risks. Legacy systems are often large, multiyear systems with complex interdependencies, and also critical to their owners. Unless the way the organization do business has significantly changed, typically the business rules encapsulated by their legacy programs are still valid and are critical for their day-to-day operations. Therefore, mainframe migration projects must be effectively managed around a solid management framework to ensure success and to mitigate risks. The main difference of migration projects from new development projects is that the existing system with ongoing typically mission critical, business applications are about to be replaced with a new system (re-engineered, redeveloped or rehosted) with no or minimal disruption to the existing system, operation or shortly business. For the development of new features or new applications compared to this situation is rather very simple from this perspective that only the newly added applications, modules or features are introduced to the system. Therefore, following a systematic, controlled and incremental project management shall form the basis of methodology.

Following is the list of the most important considerations for successful delivery of migration projects together with the use of this methodology:

- Systematic and disciplined project management is definitely number one issue. Since these projects require understanding of both an existing many-years-old existing system and expertise in mapping that to a new architecture without sacrificing any performance or availability of the ongoing business of the customer, it is a high risk if not handled correctly and carefully. More than being a technical exercise, its success depends also on the effective management of the whole life cycle and processes within which Communication management, CM and QA, Risk management plays the most important roles.

- Within the framework, in order to stay focused to the mainframe migration projects, the standard or indifferent tasks of the project management or supporting processes are not described in full detail. For example the typical tasks of project initiation (project team setup, project collaboration environment set-up, contractual preparations, organization of kick off or progress review meeting, etc) are not different from typical IT projects' processes. Therefore the details of these routine project management tasks are not defined or explained here in detail. Since this framework is intended for specifically migration projects, the routine activities of the Project management or routine activities within similar supporting processes (Quality Assurance, Configuration management, Risk management) are only referenced within the "tools/methods" section. Due to the nature of migration projects, if there are unique or critical aspects that are different from usual conduct of these processes, those are specially described in "Activity description and Considerations" section of relevant activities.

- The Business Case and the Project Plan shall be very clear, objective and shall provide the metrics to measure the success of the project. Only with a sound and clearly defined list of objectives, it would be possible to set the right level of expectations and gain continuous support from project sponsor(s). Within the framework, special work products are defined to serve this purpose as mandatory items.

- The success in the early phases of analysis and design results in correct scope definition and determination of the correct strategy of implementation plan which helps preventing scope creep, and minimizes the related risks.

- The selected strategy (not only the overall migration selection strategy but also the detailed strategy of implementation that may need to be updated throughout the lifecycle. The corresponding plans shall be clear, concise and provide a staged approach for implementation due to the fact that during the migrations both the old system and the new system will be kept running concurrently.

- Disciplined application of Configuration management, Quality Management and Test Management practices is vital for success. A very well thought Migration plan with a very well crafted and extensive technical migration design might perfectly fail due to lack use of these practices during implementation and the superseding phases.

- The first interactions during initial analysis phases are very important to set a good communication channels with organizations IT personnel involved in this migration projects. Especially at these early phases, it is very important to gain their support and gain buy-in at this level also. Getting support of these people is critical in reaching system assets and supporting in understanding their requirements and determining best implementation strategy.

- The resultant architecture shall be open for future growth, and meet the needs of organization at the technical level and fully aligned with their business level requirements.

- The training of IT staff (if possible via involvement in all phases of migration) is vital for successful handover of systems and ongoing maintenance.

- Optimize development and deployment efforts and costs also by minimizing the need for use or development of adapters, bridges or

interim applications during the transition by carefully architecting the increments. This means applications chosen for the increments shall be decided considering the cohesion, dependencies and also the need for these types of adapters mentioned.

- A challenging but manageable, doable schedule shall be in place. The implementation strategy described above for the increments shall also be designed keeping in mind the objective of minimizing the time required to develop and deploy the final solution in the target platform.

- The consistency, availability, completeness and sufficiency of all work products shall be closely observed throughout the life cycle.

- The ease of use and ease of maintenance of the target system shall always be one of the key design criteria during the lifecycle.

- Management and minimization of risks at all levels and dimensions (technical, organizational, staff related, dependencies to other systems etc) shall be exercised throughout the lifecycle with periodic risk review meetings and mitigation actions taken.

- Related with this, the implementation strategy for the increments shall also take into account the minimization of the risks as much as possible.

- Make sure the user expectations are fully met (as defined in the corresponding work products and plans) in order to have ongoing support and keep the motivation high.

- Keeping the complexity at a controllable level is highly critical for this already complex effort. Therefore, the implementation strategy for the increments and overall architecture shall take into account this driver.

With these principles in mind, following is an analysis of each of the major phases of Migration Delivery Life Cycle and provides the concerns and underlining

rationale on the structuring of each phase and major outputs as summarized in Process View.

The Analysis Phase is crucial for the success of the migration effort and starts with a study that will evaluate the organization's legacy system infrastructure and applications inventory.

Detailed evaluation of the legacy system aims to:

- Understand the legacy interfaces and application,
- Identify redundancies related to the interfaces and the applications,
- Determine the functions of the target interfaces and system,
- Understand the legacy data, identify related redundancies and determine the to-be-migrated data,
- Understand and identify the interaction of legacy system with other systems.

Analysis continues with requirements elicitation and validation. Migration Requirements Specification, specifying the target system quality attributes and describing the proposed system from the user's operational perspective is prepared. It includes end-to-end operational scenarios on how the new system is to operate from a functional stand point and fulfill the needs of diverse set of users.

Managing the migration effort is a critical success factor. Based on the findings, organizational risks are identified and a management approach suitable to the organization is determined. Prototyping and piloting needs are uncovered in accordance with the complexity of the legacy environment and risk associated with the migration.

Due to mission critical nature of the legacy information systems, availability of the system is critical for the organization. Therefore, a single deployment process will create a high risk of failure. The next step of planning is to determine a roll-out strategy for the organization by dividing the migration into many steps. Each rollout focuses on migration of certain parts of the system.

Time period needed to complete the analysis and planning phases is relatively longer than other phases as it covers consequential activities affecting the entire migration effort. Key deliverables of this phase is Migration Requirements Specification and Migration Strategy documents.

Up to now, the project team have focused on high-level questions such as “Why is this solution important?”, “What is the business value?” and “What are we going to deliver?” In the Design Phase, these questions are taken to a lower level of detail, and the questions on “How will we build this solution?” and “How will we perform the data migration” is answered. At the end of the Design Phase, the project team will have a logical solution (database and application design description documents). Each organization and their legacy software architecture are unique, migration implementation procedure provides a specific blueprint for the implementation and deployment Phases. At this point, the solution is "logical" because it exists on paper or in a design tool. This logical solution together with the implementation procedures is then passed to the Implementation Phase, where it is turned into a physical solution.

Implementation phase is the point where the project team heavily uses the migration automation tools. Prior phases provides guidance to the project team for the proper usage of the migration tools without creating a chaos that will endanger the project, thus the system and the organization.

During the implementation phase, legacy applications and database migration, including testing, is performed. Migration testing is a complex, ongoing process throughout the migration of a legacy system; often testing one system/application requires involvement of another system or program. Due to mission critical nature of the legacy information systems, it is imperative that there are no inconsistencies between the output of a legacy system and that of its replacement system. The implementation phase is also where the project team creates the support documents and training content related to the target platform.

The deployment phase is concerned with the cutover from the legacy system to the target system. When dealing with legacy systems, this process must cause as little disruption to the business as possible. The deployment phase results in the orderly transition of the enterprise from dependence on the existing legacy system to usage of the newly migrated system. This is not necessarily an instantaneous

transfer, but it is a staged rollout of the new application, a gradual replacement of the current legacy system functions.

Migration involves the physical transformation of whole or partial legacy system environment (legacy application, data, and interface and system users). Usage of Migration tools is necessary, however, only after a sound understanding of the legacy system; target system development and migration can be performed with minimal technical, business and managerial risks.

Legacy System Migration can be a very expensive procedure which carries a definite risk of failure, in order to perform a successful migration a sound management framework as presented here is necessary. Within each phase of management framework, solid software/system engineering techniques must be applied.

The target system will not be in production while the legacy system is being migrated. The legacy system will remain in production during the whole migration process. Therefore, there may be ongoing enhancements and maintenance activities on the legacy system. Testing, configuration management and quality assurance plays an essential role in all tasks throughout the whole life cycle.

5.2. Analysis Phase (Legacy System Analysis)

Analysis Phase is started after the Initiation activities. The initiation activities can be performed either as part of the migration project or as a stand alone activity during the feasibility studies or evaluation of alternatives.

It is assumed that migration is already chosen as the suitable modernization approach among alternatives. Therefore, at this point, the analysis is directly intended for migration purpose.

Table 9 Analysis Phase Overview

MAIN PURPOSE OF THE PHASE: The purpose of Legacy System Analysis is to obtain an in-depth understanding of the legacy system to determine an overall roadmap for migration efforts and to document migration requirements specification after identifying and establishing the legacy baseline. The success in this phase will result in correct scope definition of the project. Unlike other IT project, the focus during the analysis phases is on the existing mainframe environment and organization's existing business process. Since the existing portfolio of these mainframes typically contains very high volume of entities (COBOL programs, JCL's, CICS, 3270 based BMS screens, reports etc), in order for a good understanding of the existing environment, these entities and the relationships / dependencies between those needs to be identified very carefully and extensively. Therefore, compared to typical IT projects, the analysis phases tend to last longer than the design phases within which the focus is on the "to-be" migrated (target) environment.
MAIN SEQUENCE OF ACTIVITIES OF THE PHASE: 1. Initial Planning and Project initiation 2. Perform organizational and technical information gathering 3. Identify Legacy Baseline (inventory of legacy system) 4. Legacy Baseline Verification 5. Review Existing Legacy documentation 6. Perform Legacy System Assessment and Graph Relations between Legacy System Components 7. Determine Legacy Data Schema 8. Determine Interaction with Other Systems 9. Assess the overall existing situation and develop implementation strategy 10. Establish Migration Requirements Specification

Table 9 (continued)

MAIN INPUTS OF THE PHASE

1. Preliminary Project Plan (if available as an outcome of the initial activities for Migration Feasibility Study or the Modernization strategy Selection)

2. Modernization strategy Selection Report (outcome of the Modernization strategy Selection)

MAIN OUTPUTS OF THE PHASE

1. Migration Requirements Specification

Implementation Strategy Document and Project Plan with its annexes

MAIN TOOLS / METHODS OF THE PHASE

1. Migration Questionnaire

2. Mainframe Application Portfolio Assessment/Analysis Tools

3. Software Engineering and Project Management

KEY CONSIDERATIONS OF THE PHASE

The ultimate aim in this analysis phase is to have complete understanding of the mainframe environment in order to correctly plan migration phases and also to design the target environment accordingly.

As mainframe migration projects are typically change projects affecting many aspects, special consideration shall be given to identification of the change in the following domains:

1. Business process. The business process domain focuses on the business processes in which the customer act, how activities are carried out and in what sequence, what rules are followed, and the type of results obtained. Change in the business process domain is often a key driver for change in all domains.

2. Organization. The organization domain focuses on the people and organizations involved in the change: their culture, capabilities, roles, team structures, and organizational units.

3. Location. The location domain focuses on where the customer conducts business. This applies to physical facilities where people and technology reside, and to location types, such as a branch office or data center, including logical addresses such as user IDs.

4. Data. The data domain focuses on the content, structure, relationships, and business rules for the data used by the business processes, applications, and organization. It also considers the transformations needed to result in information and knowledge that the customer can use.

5. Technology. The technology domain focuses on the hardware, system software, and communications infrastructure used to enable and support solutions and services.

6. Application. The application domain focuses on the capabilities, structure, and user interface of software applications and application components used to support the change..

5.2.1. Main Activities

As it is listed in Table 9, the analysis phase consists of ten main activities. These activities are described in further detail below:

1. Initial Planning and Project Initiation

- PURPOSE:

i. Perform the routine tasks of project initiation (project team setup, project work and collaboration environment set-up, contractual preparations, organization of kick off or progress review meeting, etc). The details of these routine project management tasks are not defined or explained here in detail. Since this framework is intended for specifically migration projects, the routine activities of the Project management or routine activities within similar supporting processes (Quality Assurance, Configuration management, Risk management) are only referenced as is. Due to the nature of migration projects, if there are unique or critical aspects that is different from usual conduct of these processes, those are specially described in activity description and considerations section of relevant activities.

ii. Development of preliminary project plan. (If this draft plan is prepared during the migration feasibility/alternatives evaluation or migration strategy selection studies, a revisit or update of the preliminary project plan is performed). As a minimum the plan shall include the following items:

iii. Goals and objectives of the Organization

iv. Priorities of the Organization

v. Inputs from the Modernization strategy Selection Report (identifying the rationale in the selection of the strategy that is aligned with the business needs)

vi. High Level technical migration approach selected

vii. Project Organization, with roles and responsibilities

viii. Initial Risk list and risk management methodology that will be applied

ix. Other subsidiary management plans as annexes of the plan (QA Plan, Communication Plan, Initial Test and Evaluation Plan, etc)

- INPUTS:

  i. Modernization strategy Selection Report,

  ii. Draft project plan (if available from early feasibility or strategy selection studies)

- OUTPUTS: Initial Project Plan with annexes.

- TOOLS / METHODS:

  i. Project Management and scheduling tools

  ii. Project Plan templates

  iii. Risk management

  iv. QA/CM management

  v. Peer Review

- ACTIVITY DESCRIPTION and CONSIDERATIONS:

  i. From the project management view, the project lifecycle is started with an initial planning phase as typical for other IT projects. At this phase the information in hand can be very limited or lengthy depending on the extensiveness of the previously performed tasks that was resulted as a GO decision for the migration project. It is assumed at this phase the critical feasibility decision to migrate has already been given, and at least, the high level modernization strategy selection is already in place. Based on these assumptions, it is expected that the

project GO decision for the project is already given and the analysis phase and the following phases are intended for performing the migration and not to question or justify the decision for migration.

ii. At this stage, the main output of this activity is the Initial Plan which, due to the nature of migration projects, will be containing high level information, some assumptions and estimations about the scope of the project, size of the project and the timeline of the project. This initial plan needs to be extensively updated with the precise and detailed information gathered with the completion of the analysis phase.

iii. At the early stage, the initial plan is supposed to play a bridge role for the Organization to define what is expected from the migration project in terms of solving the business needs and addressing the underlying rationale and motivation for the migration project. The initial risk list will also be an instrumental part of the document. This way it will be ensured that the project's overall objectives are reflected to the project documentation and also the whole project team within this master document.

iv. In order to have an agreed schedule in hand together with a correct scope definition, throughout the lifecycle, it is suggested that these plans shall be constantly revisited and kept always up-to-date.

v. The list of the stakeholders of the project, within the initial project organization together with the defined roles and responsibilities is also an essential part of the initial project plan.

2. Perform Information Gathering

- PURPOSE: Obtain an understanding of the environment including: technical legacy system baseline, environment, infrastructure, IT staff resources, user profiles, training needs organizational expectations,

current state and problems of the existing system, technical and business priorities of organization, expectations from the target system, and risks.

i. For the sake of completeness, in order to gather all the required information with a systematic and documented approach, a questionnaire shall be filled together with the required stakeholders. It is not suitable to get those questionnaires without an interaction with them. It is more effective to the filling of the forms in the forma of an interview to avoid misunderstandings, miscommunications or subjective evaluations.

ii. Actual gathering of the technical assets (legacy portfolio) from the mainframe environment will be performed in the later activity (Identify Legacy Baseline) preferably with the aid of Assessment and Portfolio Analysis Applications designed for this purpose. So the main focus on this questionnaire form is on other organizational aspects.

- INPUTS:

  i. Initial plan,

  ii. Results of Initial questionnaires (if available),

  iii. Reviews,

  iv. Modernization strategy Selection Report

- OUTPUTS: Migration Information Report

- TOOLS / METHODS:

  i. Migration Questionnaire,

  ii. Migration Scorecard

  iii. Risk List

  iv. Interviews with all stakeholders

v. Peer Reviews

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. Key organizational stakeholders shall be involved when filling the forms together with the guidance of the migration experts.

ii. Migration expert shall ensure that organizations' declarations are accurate and precise.

iii. Formal meeting shall be scheduled on site for filling the forms.

iv. Migration Information Report shall be prepared by the Migration expert and shall be provided to whole Migration Team.

v. Make sure that the entire inventory items are provided by customer to prevent surprises (or scope creep) later on after the scoping is completed, or yet worse at implementation phases or after.

vi. Cross checking of the inventory items both at development and production environments helps determine actual inventory items in use and discard those obsolete and unused items (like programs, db tables batch programs and JCL's prepared for test purposes and left there after or newer versions are available). This will prevent addition of irrelevant items to the scope of migration.

vii. These first interactions are very important also to set a good communication channels with Customer IT personnel involved in this migration projects. Especially at these early phases it is very important to gain their support and gain buy-in at this level.

viii. The Questionnaire form, as a minimum, shall have the following sections:

1. general migration questions

a. objectives of the organization

b. prioritization of these objectives

c. expected risks

d. any previous migration experiences and status within the organization

e. Perceived concerns about the migration (availability, scalability, performance, security, database, 3rd party tools, batch processes and performance, IT staff availability or training needs, reliability/business continuity, SLA's)

f. Expectations about the scope of migration (whole or partial)

g. Expectations regarding the target environment and way of operation.

2. Business Needs

a. Main problem areas with the existing mainframe environment

b. ROI and TCO analysis (if performed before for the migration)

c. Business Drivers for migration

d. Expectations (short, mid or long term)

e. Perceived pros and cons of migration

3. Budget & Schedule

a. Expected timeframe for the migration

b. Migration schedule constraints (like the a specific date of business regulations, expiry date of mainframe licenses, etc)

c. Budget and budget allocation status

4. Stakeholder Analysis

a. System users and roles.

b. Total number of system users (end-user, central or field personnel)

c. IT staff (system or software).total number of system engineers, developers, Database specialists, maintenance personnel and technicians)

d. Other key stakeholders affected (integrated systems within the organization or outside, governmental agencies, partners, subcontractors personnel etc)

5. Existing Mainframe environment

a. Type and model of mainframes in the organization

b. Size of the system (MIPS count)

c. O/S of host environment

d. Utilization usage and scenarios for the CPU's.

e. Estimated number of daily transactions

f. Availability of system documentation

g. Business Processes and functionality provided

h. Availability/reliability/maintainability/performance issues with the system

i. Any other technical problems with the current system

j. Database characteristics (size, volume, data module, schemas etc)

6. Architecture

   a. Middleware components

   b. Interfaces to other systems

   c. Disaster recovery

   d. JOB Scheduler

   e. Availability of documents and knowledge for the design and architecture of the system

   f. Separation of layers within the system architecture

   g. Major design drivers and constraints

7. Applications

   a. Category of applications.

   b. Application Profiles (line of code for applications, BMS maps, JCL's, copybooks, online/batch programs, VSAM files, other file structures if exist like Sequential, indexed etc, data structure, DB details and other applicable)

   c. Primary languages used (other than COBOL like assembly or 3^{RD} or 4^{TH} level languages

   d. Operational usage scenarios for the applications (daily, day time, nightly, weekly, quarterly etc)

e. Interface to non-mainframe environments

f. Dialects of COBOL used.

g. Special usages within COBOL programs (like report writer, SORT tools etc).

h. Availability of coding standards within applications

i. Healthy, systematic Configuration management practices in place

j. Coding styles and style guides used.

8. Files

a. VSAM (size and number of files, Variable length VSAM files,

b. GDG (Generation Data Groups)

c. QSAM/BDAM files

d. Other formatted files used

9. CICS

a. Version, CICS interface used (BMS, 3270)

b. Usage statistics

10. BATCH

a. Size of batch jobs,

b. Statistics on usage and performance

c. SORT batches

d. Utilities within JCL's

11. UTILITY PROGRAMS (IDCAMS, IEBCOPY, etc)

12. PC/Workstation characteristics

13. Development Environment Details (including QA/CM and CASE tools)

14. Production Environment Details

15. IT-infrastructure

    a. Security system used (RACF, CA-TOPSECRET, etc)

    b. Usage and access method to security utilities within applications

16. Network Infrastructure

3. Identify Legacy Baseline

- PURPOSE: Establish Initial Legacy Baseline subject to analysis to determine scope (and to prevent potential scope creep)
- INPUTS: Legacy System Files and Documentation
- OUTPUTS: Initial Legacy baseline CD/DVD
- TOOLS / METHODS:

  i. Migration Library Update Reports

  ii. Mainframe Inventory Capture Tools (available from several vendors).

  iii. Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

  i. Based on the results of the questionnaire, all the inventory of existing system items subject to migration are collected.

  ii. There are two methods to identify and capture inventory:

1. Manual retrieval of all inventory items thru file transfer mechanisms like FTP.

2. Automated capturing of inventory items through "Mainframe Inventory Capture Tools" that are available from different vendors. Although this is a cost item, the advantage of this approach is not only time but avoidance of human errors.

iii. Configuration Management Specialist, together with the end user technical expert shall prepare a CD/DVD containing all legacy files (BMS Maps, JCL's, Copybooks, online/batch programs, VSAM files, data structure and other applicable)

iv. Baseline CD/DVD shall include a Configuration List.

4. Legacy Baseline Verification

- PURPOSE: Ensure complete system coverage by comparing legacy program usage statistics to the baseline CD/DVD.
- INPUTS: Legacy baseline CD/DVD
- OUTPUTS: Verified/approved Baseline CD/DVD and configuration list.
- TOOLS / METHODS:

  i. CICS Transaction System

  ii. Daily Program Usage Statistics

  iii. Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

  i. Migration Expert and migration management specialist shall receive daily program usage statistics from the mainframe and match them to the baseline CD/DVD.

ii. Following the comparison, the organization's responsible, shall formally approve configuration list and baseline CD/DVD.

iii. The typical problems in this activity occur in identifying the latest versions of the applications, especially if the configuration management practices are not in place. Typically there can be several sets of different versions of the same applications in different folders for production. Another problem area is that the valid source code of the production applications can not be reached in the corresponding source code libraries. This problem arises due to the fact that the some developers over the years tend to keep the source code modules in their own folders or project folders and there may not even be an existing test bed for the final testing of applications before putting in the actual production environment.

iv. Some of these errors can be identified during the later phase of "Perform Legacy System Assessment and Graph Relations between Legacy System Components" while checking the dependencies and interrelations of the modules (with analysis tools available.

v. In both problems (duplicated code or missing code) it severely affects the actual scoping of the project. Another side effect would be in the area of architecting applications in the target environment.

vi. Another typical problem is the identification of applications without a source code. Over the years some of the source code might be lost and only the executable files for those applications can be available. Special consideration shall be given to this type of problems.

vii. To identify these problems Load Modules listings available within the system and similar features available in the CICS

transaction system side can be used to trace the actual production applications and the corresponding source codes.

viii. Special care shall be given to applications that are only executed seasonally or once a year (which are typically found in reporting applications or budgetary system applications). These applications will not be listed in the above mentioned load list or load libraries on the system expect some special dates. The questionnaire filled in early phases shall address the program usage scenarios to avoid this problem.

5. Review Existing Legacy documentation

- PURPOSE:

  i. Determine and capture legacy documentation (including user manuals, operating manuals, design documents etc).

  ii. Get an understanding and functioning of legacy applications, identify redundancies and determine potential functions of the target applications through informal review of the legacy system documentation.

- INPUTS: Legacy documentation

- OUTPUTS:

  i. Existing Legacy Documentation List

  ii. Legacy Documentation Review Test/Report

- TOOLS / METHODS:

  i. Configuration Management

  ii. Peer reviews

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

  i. When performing informal reviews, reviewers should try to identify the possibility to use the user manuals and design

documents, as a guideline for further phases (for example: for functional test procedures that will be used during unit and integration testing).

6. **Perform Legacy System Assessment and Graph Relations between Legacy System Components**

- PURPOSE:

    1. Increase the understanding of legacy applications and their interrelations by automated review and analysis of the complete inventory at all levels.

    2. Provides insights that help the migration team in subsequent migration design and increment planning activities (especially via the interdependency diagrams of modules)

    3. Determine #of files, #of programs, #of copybooks, #of tables, #of statements, #of BMS, #of JCL's, line of code, variables, inter-relations and dependencies among above. Legacy data and database is analyzed in another separate activity.

- INPUTS:

    i. Legacy Documentation Review Test/Report

    ii. Legacy Baseline CD/DVD

- OUTPUTS:

    i. Legacy System Assessment Report

    ii. Initial Configuration Item List

    iii. Legacy System Interdependency Diagrams

- TOOLS / METHODS:

i. Mainframe Application Portfolio Assessment/Analysis Tools (available from several vendors)

ii. IT Application Portfolio Analysis Methods and practices

iii. Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. Some analysis tools were prepared and proposed within the academic research activities for similar purposes. But the vast majority of analysis tools are offered by Migration Tools vendors like Micro Focus, Relativity, EZ Legacy, MetaWare, and BluePhoenix etc.

ii. Typical features these tools can provide during the migration are as follows:

1. The application artifacts, like COBOL source files and copy books, JCL sources and BMS/MFS screens, are loaded into the project and a complete application inventory is constructed in a project view.

2. Data flow analysis.

3. Support for many standard mainframe APIs like EXEC CICS, EXEC DLI, CBLTDLI and EXEC SQL.

4. All application components and the interrelations between them including screen and data access is presented.

5. Any missing components or unused components are identified.

6. Documentation and many complicated reports can be generated as a result of the analysis from different views like usage of programs in the subsystems/modules, usage of variables, database calls, security module calls, program variables in each

program, call to other programs, external systems, CICS Calls, dependency reports for each program or module etc are among those reporting features.

7. Some of the tools also offer business rule mining and identification of business processes from the source code especially for the re-use potential within SOA environments.

iii. This information increases the understanding of the online and batch elements of the system, programs, copybooks, screens, databases, files, variables, paragraphs and statements.

iv. With these in hand, Migration team can perform impact analysis to automatically assess the ripple effect of a potential modification throughout the application before making the changes which rework and correspondingly the overall migration life cycle.

7. Determine Legacy Data Schema

- PURPOSE: Understand the legacy data and databases and identify redundancies for scoping Data Migration.
- INPUTS: Legacy Database and files (VSAM, ISAM, etc)
- OUTPUTS:
  i. Legacy System Data Schema & Analysis Report
  ii. Data Migration Scope
- TOOLS / METHODS:
  i. Data Analysis Tools
  ii. Mainframe Application Portfolio Assessment/Analysis Tools (available from several vendors)
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. This activity can be performed mainly in two phases:

1. Identification of the data and database schema from the Database definition and table definition files from the used databases (typically DB2) definition.

2. Through getting a sample database which is large enough to provide representative set of legacy database.

3. It is also possible to face the situation that the database is already migrated to non mainframe based platforms already. Also in some migration projects it is common to see that data migration is not planned and the database stays within the mainframe environment.

4. Data migration can be performed within the overall migration project or can be independently planned as a separate initiative. Therefore within the lifecycle, the data migration subject is not extensively studied and only the main considerations critical to the overall migration is emphasized throughout the lifecycle.

8. Determine Interaction with Other Systems

- PURPOSE: Identify and understand interaction and dependencies (hardware, software and data communications) on external systems.

- INPUTS:

  i. Legacy System Assessment Report

  ii. Legacy Initial Configuration Items List

  iii. Declarations received from IT staff based on interviews and peer reviews

  iv. Legacy baseline CD/DVD

- OUTPUTS: Legacy System Assessment Report (update)

- TOOLS / METHODS:

    i. Formal Meetings with IT Staff and possible user groups

    ii. Manuals and Network Diagrams within the existing system documentation

    iii. Legacy Site Survey.

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

    i. Migration team shall not only identify the interactions with other systems but also try to determine the possible effects of migration effort to those systems.

    ii. The results of this activity should be very well documented and reviewed by all stakeholders in order not to miss any of the interfaces also to be taken into account for the development of modernization strategy aligned with business objectives.

9. Assess the overall existing situation and Develop Implementation Strategy

- PURPOSE:

    i. Based on the entire artifacts collected do an analysis of all situation preferably in a session to which whole migration team participates. Assess the current state of the project and perform a gap analysis to determine the desired state of the project and target environment.

    ii. Determine high level migration solution based on the selected modernization strategy, analysis findings and the above mentioned situation analysis meetings.

- INPUTS:

    i. All outputs from previous activities

    ii. All outputs from pre-project phases:

1. Preliminary Project Plan (if available as an outcome of the initial activities for Migration Feasibility Study or the Modernization strategy Selection)

2. Modernization strategy Selection Report (outcome of the Modernization strategy Selection study – if available)

- OUTPUTS:

    i. Project Plan and annexes (update)

    ii. The results of this activity can be documented either as a single "Implementation Strategy" document or can be separated to the following documents:

        1. Modernization strategy Selection Report (update)

        2. Specific Migration Life Cycle Model (update: Customizations/Adaptations on this Migration Lifecycle Model of the framework)

        3. Data Migration Strategy & Scope

        4. High Level Deployment Strategy. (with draft planning of Increments)

- TOOLS / METHODS:

    i. Mainframe Application Portfolio Assessment/Analysis Tools

    ii. Systems Engineering

    iii. Software Engineering

    iv. Project Management

    v. Decision Making Techniques (Wideband Delphi, etc.)

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. The intent of this activity is not the selection of overall migration strategy. Throughout the life cycle it is assumed that the overall strategy was already given in pre-project phase where a decision for the project's realization is given. Therefore, aligned with the overall migration decision (migrate/rewrite/replace/do-nothing), this activity aims at developing the high level implementation strategy of the selected migration approach (re-host/language conversion/hybrid). Within this activity, the following factors shall be covered in the corresponding output work products:

    1. Migration Domain (objective, business case, time, budget, risks, etc.)

    2. Complexity of migration.

    3. Needs and priorities (Partial or complete migration, partial or complete data migration, need for new functionality)

    4. Risk management (technical and contractual)

    5. Available resources and competencies

ii. Update of Project plans and other administrative documents. The resultant Implementation Strategy shall be used as a major input for the development of managerial documents (SOW, Contract if applicable, project plans etc. if not already prepared) and as a guide for supporting managerial decisions throughout project delivery. Otherwise, it shall be used for revising those documents accordingly.

iii. This high level document will also serve another purpose that is the development of Migration Requirements Specification document in subsequent activities. A detailed version of this "Implementation Strategy" document will be prepared in Design Phase (Migration Implementation Plan).

10. Establish Migration Requirements Specification

- PURPOSE: Combine and document outputs of analysis activities to establish a technical baseline that will be a guideline for forthcoming delivery phases in requirements specification format.
- INPUTS:
    i. Migration Information Report
    ii. Implementation Strategy Document
    iii. Legacy Documentation Review Test/Report
    iv. Legacy System Assessment Report
    v. Legacy System Interdependency Diagrams
    vi. Legacy System Data Schema & Analysis Report
- OUTPUTS: Migration Requirements Specifications (MRS)
- TOOLS / METHODS:
    i. Requirements Engineering
    ii. Software Engineering
    iii. Peer Review
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. MRS contains the needs of each stakeholder, sponsor, end users and IT staff (that will mange the systems after migration) as a basis of requirements.
    ii. MRS is subject to formal review by all project stakeholders.
    iii. MRS is submitted to the sponsor and decision making authority for formal approval and acceptance.

MRS is subject to Configuration Management throughout the subsequent phases of life cycle which means it should be kept up-to-date and the requirements stated in the documents shall be traced within the subsequent documents.

5.3. Design Phase (Implementation Planning)

Table 10 Design Phase Overview

MAIN PURPOSE OF THE PHASE: The purpose of the design (implementation planning) phase is to document in detail how the legacy system will be migrated to the target environment based on the Migration Requirements Specification and to provide a technical plan (migration implementation plan) that will set the road map of the technical implementation in subsequent phases. The implementation strategy that is prepared in the Analysis phase will be used as the draft or initial of this technical plan. The migration team transforms the requirements into an initial concept of how the solution in target environment solves the business problem. The solution concept serves as a baseline and sets the stage for the more formal design of the solution. On the managerial side, the Project Plan and its annexes will be update provide a guideline to determine the progress of the project at specified milestones based on quantifiable measures.
MAIN SEQUENCE OF ACTIVITIES OF THE PHASE: 1. Determine the target architecture 2. Perform Target Data and Database Design 3. Determine Legacy Sample Data 4. Perform Trial Study 5. Prepare Migration Implementation Plan 6. Update Project plan and its annexes
MAIN INPUTS OF THE PHASE 1. Migration Requirements Specification 2. Implementation Strategy Document 3. Project Plan and its annexes

Table 10 (continued)

MAIN OUTPUTS OF THE PHASE

1. Migration Implementation Plan
2. Project Plan and its annexes (update)

MAIN TOOLS / METHODS OF THE PHASE

1. Migration Toolkit (based on the chosen migration approach)
2. Development/test environment for Trial or Pilot Study
3. Software Engineering
4. Project Management

KEY CONSIDERATIONS OF THE PHASE

During the design, the project team (including the members from the organization) will establish a logical solution - target architecture diagrams and database design description documents. Next step of design phase is to determine a roll-out strategy for the organization by dividing the migration into many steps. Each rollout focuses on migration of a certain part of the target system. Migration Implementation Plan is a technical document describing how the legacy migration will be performed. Further analysis and design related to rollouts (increments) will be detailed during the implementation phase.

Legacy system analysis and design phases are relatively more challenging than other phases as it covers consequential and comprehensive activities affecting the entire migration effort.

At this point, the solution is still "logical" because it exists on paper or in a design tool. This logical solution described in detail in the Migration Implementation Plan is then passed to the Implementation Phase, where it is turned into a physical solution.

The Analysis Phase was heavily focused on the client and stakeholders. The Design Phase is focused on the project team and includes the work required to design the solution. There are an infinite number of potential solutions. The Design Phase is where the project team determines the best solution possible, given the best requirements and the architecture and standards that already exist in the environment.

5.3.1. Main Activities

As it is listed in Table 10, the analysis phase consists of six main activities. These activities are described in further detail below:

1. **Determine target architecture**

   - PURPOSE: Develop the blueprint of the target system at the highest level of components and their most important interactions. Conceptual logical and physical designs are workout to a certain detail at this activity.

   - INPUTS:

     i. Migration Requirements Specification

     ii. Implementation Strategy Document

     iii. Legacy Data Schema

     iv. Project Plan and its annexes

   - OUTPUTS: Target System Architecture Diagrams

   - TOOLS / METHODS:

     i. Software Engineering

     ii. Systems Engineering

     iii. UML

   - ACTIVITY DESCRIPTION AND CONSIDERATIONS:

   - As part of the document the architecture (the physical view of the target system showing the mapping of software to hardware), the logical view and the process model (concurrency and synchronization aspects) shall be included.

- Target System Architecture Diagrams are subject to formal review.
- Quality Attributes (Performance, availability, scalability, redundancy etc) shall be the primary driver for target architecture determination.

2. Perform Target Data and Database Design

- PURPOSE: Develop the target data schemas and determine the mapping rules.
- INPUTS:
    i. Legacy Data Schema
    ii. Legacy Data
- OUTPUTS:
    i. Database Design Descriptions (Target Data Schema/ Structure of Temporary Storages for Data Conversion/Database Mapping Rules)
- TOOLS / METHODS:
    i. Software Engineering
    ii. Systems Engineering
    iii. UML
    iv. Data Conversion Tools if needed (available from many vendors)
    v. Database Design Descriptions Template & checklists
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. Due to the diversity of the database choices (especially for the target platform databases like MS SQL server, ORACLE, etc); no specific, step by step database migration is provided for a named Database system. (Refer to Constraints section of the lifecycle). But common points of database migration that must

be taken into account and mandatory for each successful migration are taken into account within the scope of this lifecycle.

ii. Special consideration shall be given to VSAM, ISAM and sequential files. The VSAM, ISAM or other file systems migration can either be performed in two methods:

1. Manual Conversion methods for the files that can be developed specifically during the migration.

2. IF the size and type of data files are lengthy, it is also possible to use data file conversion (VSAM, ISAM, IMS etc) tools commercially available from many software vendors specifically designed for the data migration tasks.

iii. Quality Attributes (Performance, availability, scalability, redundancy etc) shall be the primary driver for target architecture determination.

iv. Database Design Descriptions should contain information related to Storage Requirements for Data Migration, Software Requirements for data migration, Hardware Requirements for data migration, and Resource requirements for data migration.

3. Determine Legacy Sample Data

- PURPOSE: Determination of datasets to be extracted from legacy (databases, tables, data structures, relations and content) to be used during development and test activities

- INPUTS:

i. Legacy Data Schema

ii. Database Design Descriptions Document

- OUTPUTS:

i. Database Design Descriptions (Target Data Schema/ Structure of Temporary Storages for Data Conversion/Database Mapping Rules)

- TOOLS / METHODS:

  i. Database Design Descriptions Template & checklists

  ii. Data Analysis tools (depending on the target database capabilities)

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

  i. Analysis of the Legacy data to identify the sample legacy data set is very critical. Sample data set should have the same structure (or content) with the real data set that will be used after deployment. Early detection of probable problems caused by data structure and data inconsistency is another important issue.

  ii. Problematic data format and contents should be identified and considered in sample data set.

  iii. Data migration strategy and target data architecture, especially data that will be kept in Legacy environment and data that will be converted and used from the target platform shall be considered.

  iv. Development and test data sets should be consistent.

4. Perform Trial Study

- PURPOSE: Obtain better understanding of the target system design and the migration effort ahead.

- INPUTS:

  i. Legacy Programs

  ii. Legacy Sample Data

- OUTPUTS:

    i. Proven Solution/Concept

    ii. Trial Study Report

    iii. Migration Implementation Guideline (draft)

- TOOLS / METHODS: Based on the chosen migration approach the tools / methods vary. For example if a re-host approach was chosen than the corresponding toolset of that re-host platform is used. Or if language conversion approach is chosen than this trial study involves usage of corresponding language conversion tools.

    i. Migration Tools based on the approach chosen as described above.

    ii. A sample Target System environment developed for this trial study or pilot.

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

    i. Depending on the chosen migration approach, the content and processing of the trial study varies. But from the overall technical management perspective, conducting a lessons-learned meeting following the trial study is essential.

    ii. The results of the trial study and the recommendations for future migration procedures shall be well documented as a guideline for subsequent phases.

    iii. Based on the trial study results, the target system architecture diagrams and related design documents are updated if necessary.

5. Prepare Migration Implementation Plan

- PURPOSE: Design and document the migration (conversion, integration, deployment, operations, V&V) cycles in detailed definitions including answers of “how’s” as an essential

implementation reference in harmony with the managerial project plans

- INPUTS:
    i. All the output generated within this phase
    ii. MRS
- OUTPUTS:
    i. Migration Implementation Plan
- TOOLS / METHODS:
    i. Systems Engineering
    ii. Software Engineering
    iii. Peer Reviews
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. The migration implementation planning is one of the synthesis activities that is bringing together all the collected information from analysis and transforming it to the design of the target architecture and solution based on the requirements, vision and scope.
    ii. From a technical standpoint, the following aspects of the target environment and architecture shall be defined at the end of this activity:
        1. Architecture
        2. User Interfaces
        3. System Documentation
        4. Infrastructure components and services
        5. Communication and network model

6. Security and other peripheral services
7. Architectural and operational constraints
8. Data models
9. Interfaces to other systems, agencies or services, interoperability considerations
10. Quality of Service considerations
11. Service Execution Platform and Service Management
12. Problem and Incident Management procedures
13. Support Services
14. Operations and logistics

iii. Similarly the following points should be identified during this activity:

1. Is the selected technology of the target system mature and stable enough?
2. Is there a road map available for the chosen technology?
3. Does the trial or pilot study of migration justify the benefits of the chosen technology and architecture?
4. Is there (or going to be) any compatibility issues observed?
5. Are there any dependencies of the adopted technology?
6. Is the impact of application of the adopted technology feasible (in terms of cost, schedule etc)?
7. What are the training requirements and is it available?

8. What is the typical learning curve for the current IT staff to adapt to the new technology?
9. What testing will be needed?
10. What is the disaster recovery features?
11. What is the system availability practices available
12. Planning for “Cut-over” considered?

iv. Based on the above given checklist, at the end of the preparation of the plan, the migration implementation plan should contain as a minimum the followings:

1. Legacy Data Schema
2. Legacy Sample Data Definition
3. Target System Architecture Diagrams
4. Sequence and Scope of Increments
5. Sequence and Scope of adapters to be generated or used.
6. Increment schedule
7. Resource planning
8. Development Environment Specification
9. Test Environment Specification
10. Production Environment Specification
11. Data Conversion Environment Specification
12. Data Conversion Plan
13. Database Design Descriptions
14. Migration Implementation Guideline

15. Deployment Plan

16. Operations Plan.

17. Test and evaluation plans

18. Trainings needs

v. This plan should give detailed info about the expected completion level, maturity and success measures of migration implementations in specified milestones.

6. Update Project plan and its annexes

- PURPOSE: The project plan and its annexes should be updated to reflect the latest artifacts and status after the design activities before the start of mass migration tasks starts.
- INPUTS:
    i. Initial Project Plan with annexes
    ii. Migration Implementation Plan
- OUTPUTS: Project Plan with annexes
- TOOLS / METHODS:
    i. Project management and scheduling tools
    ii. Systems Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. The initial project schedule is also reorganized based on the chosen implementation approach for increments (or water fall). It serves as the master project plan for the roll-up of all other plans (in the annex); therefore all the strategic items, consolidation approaches, dependencies and assumptions shall be in place.

ii. Considering the duration from the first activation, the project organization, project team assignments, and any other resource assignments and the corresponding plans should also be updated with the coordination of the resources responsible.

iii. Since the actual migration environment requirements for development/test/migration environments are defined within the Migration Implementation plans, the corresponding sections and the required procurement activities for those environments shall be carefully re-planned.

5.4. Implementation Phase

Table 11 Implementation Phase Overview

MAIN PURPOSE OF THE PHASE: Implementation Phase consists of Development Environment Preparation, Increment Based Analysis and Design, Legacy System/Data Conversion, Integration and Acceptance Activity Groups.
MAIN SEQUENCE OF ACTIVITIES OF THE PHASE: 1. Development Environment Preparation 2. Analysis and Design (Increment Based) 3. Legacy Conversion (Increment Based) 4. Data Conversion (if applicable) 5. Integration 6. Acceptance
MAIN INPUTS OF THE PHASE 1. Migration Requirements Specification 2. Migration Implementation Plan 3. Project Plan and its annexes.
MAIN OUTPUTS OF THE PHASE 1. Migrated System (Target solution and Target Solution Documentation) on Target Environment 2. Project Plan and its annexes (update)

Table 11 (continued)

MAIN TOOLS / METHODS OF THE PHASE

1. Migration Toolkit (based on the chosen migration approach)
2. Development/test environment for conversion
3. Software Engineering
4. QA/CM
5. Project Management

KEY CONSIDERATIONS OF THE PHASE

The purpose of the Increment Based Analysis and Design is to obtain detailed information on legacy system components to be converted during the increment and to obtain verification references related to the target components.

Legacy Conversion, correction and verification is performed incrementally in development environment. Converted and verified components are transferred into test environment for integration and validation.

Integration is a complex, ongoing process; often integration and testing of one component requires involvement of other components and legacy system. Due to mission critical nature of the legacy information systems, it is imperative that there shall be no inconsistencies between the output of a legacy system and that of its replacement system. The implementation phase is also where the project team creates the support documents and training content related to the target platform.

Validated solution and documentation are submitted for acceptance. The purpose of acceptance testing is to obtain formal end-user approval in accordance with the project plans, prior to the deployment of the solution.

5.4.1. Main Activities

As it is listed in Table 11, the implementation phase consists of six main activities. These activities are described in further detail below:

1. DEVELOPMENT ENVIRONMENT PREPARATION

This activity is further subdivided to the following sub activities:

a. Establish Target System Development Environment

b. Establish Test Environment

c. Allocate Necessary Infrastructure for Data Conversion (when data conversion is necessary)

Below each of these sub activities are further described:

a. Establish Target System Development Environment

- PURPOSE: The purpose of development environment preparation is to allocate necessary resources and establish development, test and data conversion environments early in the project lifecycle.
- INPUTS: Migration Implementation Plan
- OUTPUTS: Target System Development Environment
- TOOLS / METHODS:
    i. Migration Tool Kit
    ii. Target System Resources
    iii. Configuration Management
    iv. Systems Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Once established the environment should be placed under configuration control. The Organization should allocate necessary resources and workspace accordingly.

b. Establish Test Environment

- PURPOSE: Establish the test environment in accordance with the Migration Implementation Plan
- INPUTS: Migration Implementation Plan
- OUTPUTS:
    i. Target System Test Environment

ii. Legacy System Access to test environment

iii. Legacy System Test Data

iv. Database with sample data set

- TOOLS / METHODS:

  i. Migration Tool Kit

  ii. Target System Resources

  iii. Configuration Management

  iv. Software Engineering

  v. Systems Engineering

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Once established, the test environment should be placed under configuration control. The Organization should allocate necessary resources and workspace accordingly.

c. Allocate Necessary Infrastructure for Data Conversion (when data conversion is necessary)

- PURPOSE: Allocate necessary resources for data migration and establish physical data migration environment.

- INPUTS: Migration Implementation Plan

- OUTPUTS: Target Server where the target database and data will reside.

- TOOLS / METHODS:

  i. Database Server

  ii. Database with sample data set

  iii. Configuration Management

  iv. Software Engineering

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Once established data migration environment is under configuration control with limited access to the server. Depending on the data conversion strategy, the legacy data will be gradually converted into the target database. Therefore the infrastructure necessary for data conversion shall be established early in the project lifecycle. Legacy data will be gradually converted and validated to this server. The data on this server must not be used for any tests. Following the acceptance of the solution, this is the database server that will go live together with the deployed solution The Organization should allocate necessary resources and workspace accordingly.

2. ANALYSIS AND DESIGN (INCREMENT BASED)

The purpose of this increment based analysis and design is to obtain detailed information on legacy system components to be migrated during the increment and to obtain verification references related to the target components.

This activity is further subdivided to the following sub activities:

a. Perform User Interface Functionality Analysis

b. Determine Adapters Specification

c. Determine Business Rules (optional)

d. Target Component Design

e. Determine and Define Data Conversion Control Reports

f. Establish User Interface Test Procedures

Below each of these sub activities are further described:

a. Perform User Interface Functionality Analysis

- PURPOSE: Obtain information on legacy system interface functionality, which will be used as a verification resource.

- INPUTS: Legacy System User Interfaces (BMS Maps)

- OUTPUTS: Updated MRS (User Interface Screen Captures/User Interface Functionality Statements)
- TOOLS / METHODS: User Interface Functionality Analysis Guideline
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Initially, target user interfaces shall provide the same touch and feel as the legacy system for ease of testing by the users. During this analysis, functionality on user interfaces that does not work as needed shall be identified. Legacy System Interface inter-relations shall be determined as part of analysis.

b. Perform User Interface Functionality Analysis

- PURPOSE: Determine adaptor specifications for establishing communications between Legacy System and Target Components.
- INPUTS:
    i. MRS
    ii. Migration Implementation Plan
- OUTPUTS: Updated MRS
- TOOLS / METHODS: Software Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Adaptors needed for both data conversion and program communications shall be considered. Ensure that each adaptor has its own unique Identification Name and Number.

c. Determine Business Rules (optional)

- PURPOSE: Obtain information on operation logic of critical legacy system programs, which will be used as a verification resource.
- INPUTS: Critical Legacy System Programs/Files
- OUTPUTS: Updated MRS

- TOOLS / METHODS: UML or other representation tools or methods
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Legacy System business rules are embedded and dispersed in the code and end-users generally are unaware of them as a whole. However determining the business rules will ensure more effective verification. As a caution: if determination of business rules is a necessity, there may be an unwillingness to expose them to the migration team.

d. Target Component Design

- PURPOSE: Determine in detail the design constraints and differences between Legacy System and Target System.
- INPUTS:
    i. MRS
    ii. Migration Implementation Plan
- OUTPUTS: Component Design Description section of the Implementation Plan
- TOOLS / METHODS:
    i. UML, MS Visio or similar representation tools
    ii. Migration Tool kit
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: According to the project specifications as described in the contract and migration implementation plan target component design activity might not be applicable.

e. Determine and Define Data Conversion Control Reports

- PURPOSE: Obtain information on Legacy System operation results for the comparison and verification of Target System Components.
- INPUTS: Legacy System Reports concerning the scope of the increment

- OUTPUTS: Data conversion Test Procedures
- TOOLS / METHODS:
    i. Legacy System Reporting Tools,
    ii. Target System Reporting tools/custom applications
    iii. Migration Tool kit
    iv. Configuration Management
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: IT Staff (if not part of migration team) shall be involved in determining and obtaining the data conversion control reports and shall approve that reports are complete and accurate. Data Migration Control Reports shall be subject to Configuration Management.

f. Establish User Interface Test Procedures

- PURPOSE: Gather the outputs of analysis and design activities and establish a test procedure to be used for the verification of the components.
- INPUTS:
    i. MRS
    ii. Migration Implementation Plan
    iii. Data Migration Test Procedures
- OUTPUTS: User Interface Test Procedures (for unit testing)
- TOOLS / METHODS:
    i. Peer Review
    ii. Software Engineering
    iii. Configuration Management

ACTIVITY DESCRIPTION AND CONSIDERATIONS: Initial User Interface Test Procedures are subject to peer-review and configuration control.

3. LEGACY CONVERSION (INCREMENT BASED)

The purpose of the Incremental Legacy Conversion is to incrementally convert the legacy system in order to mitigate risks and provide timely feedback on conversion process.

This activity is further subdivided to the following sub activities:

a. Establish Legacy Baseline related to the Increment

b. Perform Increment Implementation Readiness Audit

c. Perform Conversion of the Legacy System Components

d. Perform Increment Tweaking

e. Develop/Test Adaptors (when necessary)

f. Perform Code Review

g. Unit Test the components

h. Create Conversion Documentation

i. Create Training Content

j. Document Lessons Learned

Below each of these sub activities are further described:

a. Establish Legacy Baseline related to the Increment

- PURPOSE: Update legacy system baseline subject to verification
- INPUTS: Legacy System
- OUTPUTS: Legacy System Baseline
- TOOLS / METHODS: Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Once the Legacy System is base lined for conversion, any changes to the legacy system must not be allowed. Request for critical changes shall be subject to impact analysis and Configuration Control. All changes to the Legacy System components that are converted shall be documented, reviewed and approved by Migration team.

b. Perform Increment Implementation Readiness Audit

- PURPOSE: Validate that previous activities are completed /performed adequately and check that the legacy system components that will be converted are base lined
- INPUTS:
    i. All Increment Related Outputs.
    ii. Legacy System Baseline
- OUTPUTS: Increment Implementation Readiness Audit Report
- TOOLS / METHODS:
    i. Software Quality Assurance
    ii. Implementation Readiness Audit Checklist
    iii. Project Management
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: An independent Specialist (Quality Assurance) shall perform increment Implementation Readiness Audit. The results of the audit shall be reported to the Migration Project Manager. The project team shall be informed of the audit results.

c. Perform Conversion of the Legacy System Components

- PURPOSE: Establish Target System incrementally
- INPUTS: Components of the Legacy System that will be converted

- OUTPUTS: Target system component
- TOOLS / METHODS:
    i. Migration Tool Kit
    ii. Software Engineering
    iii. Configuration Management
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: The migration shall be well informed on the conversion tools prior to this activity. The migration conversion team shall document the problems faced during conversion in order to improve conversion process.

d. Perform Increment Tweaking

- PURPOSE: Fix the converted components where the migration tools fall short.
- INPUTS: converted components
- OUTPUTS: fixed converted components
- TOOLS / METHODS:
    i. Software Engineering
    ii. Configuration Management
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: It is certain that even if a tool based migration process is chosen there will be areas where manual intervention will be essential. Depending on the unique situation of the organization this activity will vary a lot for the specific cases and even between increments.

e. Develop/Test Adaptors (when necessary)

- PURPOSE: Develop adaptors to establish communication between Legacy system and target system components and verify that the adaptors serve the intended purpose.

- INPUTS:
    i. MRS (adaptor specification)
    ii. Migration Implementation Plan
- OUTPUTS: Adaptors
- TOOLS / METHODS:
    i. Migration Toolkit (if applicable)
    ii. Target System tools (if applicable)
    iii. Software Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Integration issues should be considered during the development of adaptors

f. Perform Code Review

- PURPOSE: Ensure that conversion is performed in accordance with the migration implementation plan and guideline and there are no inconveniencies in migrated code.
- INPUTS:
    i. converted legacy system components
    ii. adaptors
- OUTPUTS:
    i. reviewed and updated converted components and adaptors
    ii. Review/test report
- TOOLS / METHODS:
    i. Software Engineering
    ii. Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: An independent reviewer or tester should perform the review.

g. Unit Test the components

- PURPOSE: Ensure that converted components and adaptors perform their intended tasks.
- INPUTS:
    i. Reviewed, updated and converted components and adaptors
    ii. MRS
    iii. Initial User Interface Test Procedures
- OUTPUTS:
    i. Tested, updated and converted components and adaptors
    ii. Review/test reports
- TOOLS / METHODS:
    i. Migration Toolkit
    ii. Target Environment Test tools
    iii. Software Engineering
    iv. Configuration Management
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: An independent tester should perform the unit tests.

h. Create Conversion Documentation

- PURPOSE: Collect and summarize the efforts of increment and prepare manuals concerning the target system.
- INPUTS: All increment related outputs
- OUTPUTS:

i. Increment summary report

ii. Draft maintenance manual

iii. Draft user's manual

- TOOLS / METHODS:

  i. Increment Summary Report Template

  ii. Maintenance Manual Template

  iii. User's Manual Template

  iv. Configuration Management

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Increment Summary Report shall, at least, document the scope, legacy system components converted, adaptors developed, code review and unit testing results, duration and cost of the increment. Maintenance Manual is prepared incrementally through the conversion process. User Manual shall be written together with the user's representatives in order to ensure ease of use and functionality.

i. Create Training Content

- PURPOSE: Prepare training materials for the customer representatives in order to ease deployment and operating issues.

- INPUTS: All increment related outputs

- OUTPUTS: Training Materials

- TOOLS / METHODS: Training Material Template

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: As the target system interfaces will have the same touch and feel as the legacy system end user training need shall be minimal. The training content should ease the transition of COBOL programmers and green screen users to new windows / browser based GUI environment. COBOL

programmers may require training and end- users may require windows / browser environment training.

j. Document Lessons Learned

- PURPOSE: Improve the target system and the migration methodology.
- INPUTS: Experiences gained during the migration processes and specific increment.
- OUTPUTS: Lessons Learned Document
- TOOLS / METHODS: Lessons Learned Template

ACTIVITY DESCRIPTION AND CONSIDERATIONS: All team members shall participate to lessons learned meeting and share their experiences. Lessons learned meeting shall be arranged at the end of the increments and lessons learned are shared with the project team. Ensure that the lessons learned are integrated back into the methodology.

4. Data Conversion (if applicable)

The purpose of data conversion is to take existing legacy database structure / data (including VSAM and sequential files) and change it into a format that can be used directly by the new or modified solution and target database.

This activity is further subdivided to the following sub activities:

a. Data Conversion Readiness Review

b. Perform Data Conversion

c. Validate Legacy Data Conversion through Control Reports

d. Baseline Target Database

e. Revise Target Database

Below each of these sub activities are further described:

a. Data Conversion Readiness Review

- PURPOSE: Review readiness for data conversion
- INPUTS:
    i. Data Conversion Plan
    ii. Data Conversion Guideline
- OUTPUTS: Audit Report / Corrective Actions
- TOOLS / METHODS:
    i. Data Conversion Checklist
    ii. Software Engineering
    iii. QA/CM Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. Review the availability of adequate resources.
    ii. Review conversion sequence steps.
    iii. Review if all COBOL programs containing SQL statements are identified.
    iv. Review if all VSAM/ISAM and sequential files are covered.

b. Perform Data Conversion

- PURPOSE: Orderly conversion of legacy data to target database.
- INPUTS:
    i. Legacy Data
    ii. COBOL source code related to SQL statements and VSAM/sequential files
- OUTPUTS:

i. Target system database

ii. SQL statements modified in COBOL source code in accordance with the requirements of the target database

- TOOLS / METHODS:

i. Target Database db import/transport tools and features.

ii. Data File converter tools (if used available from several software vendors)

iii. Data Conversion Checklist

iv. Software Engineering

v. ACTIVITY DESCRIPTION AND CONSIDERATIONS: Depending on the target database requirements SQL statements in COBOL programs shall be carefully modified if needed based on the DB system used.

c. Validate Legacy Data Conversion through Control Reports

- PURPOSE: Validate data conversion by running control reports against legacy data and comparing the results.
- INPUTS: Legacy and Target reports run in accordance with the control report specifications
- OUTPUTS:

i. SQL statements modified in COBOL programs

ii. Validated Target Data

- TOOLS / METHODS:

i. One to one comparison

ii. Legacy system reporting tool

iii. Target System Reporting

iv. Software Engineering

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Make sure that all the embedded SQL statements are replaced with the appropriate API calls of the Target database. Make sure that all the sequential and VSAM files are covered during the conversion.

d. Baseline target database

- PURPOSE: Establish a baseline for the validated target database and create a full backup for restore purposes.
- INPUTS: Validated Target Data
- OUTPUTS:
  i. Target database complete backup
  ii. Configuration Management Baseline
- TOOLS / METHODS:
  i. Target Database tools backup/restore features.
  ii. Software Engineering
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Depending on the target database requirements SQL statements in COBOL programs shall be carefully modified if needed based on the DB system used.

e. Revise Target Database

- PURPOSE: Perform model modifications, transformation rule adjustments and script modifications
- INPUTS: Validated Target System Data
- OUTPUTS: Logical and Physical Data Model Synchronization

- TOOLS / METHODS:
    i. Target Database tools
    ii. Software Engineering

ACTIVITY DESCRIPTION AND CONSIDERATIONS: N/A.

5. Integration

The purpose of Integration is to verify and validate the converted programs in test environment and their interactions with legacy system and database.

This activity is further subdivided to the following sub activities:

a. Create Integration Test Procedures

b. Establish/Baseline Sample Database for Integration Testing

c. Perform Integration Testing

d. Create Acceptance Test Procedures

Below each of these sub activities are further described:

a. Create Integration Test Procedures

- PURPOSE: Verification guideline for the correctness of interactions among programs and data.
- INPUTS:
    i. MRS
    ii. Migration Implementation Plan
    iii. Solution to be deployed
    iv. Adaptor Specifications
    v. Unit Test Results

- OUTPUTS: Integration Test Procedures
- TOOLS / METHODS: Software Engineering (Software Verification and Validation)
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Test coverage should be considered and planned in accordance with migration schedule. Some tests may be postponed into forthcoming integration efforts to minimize the necessity of adaptors.

b. Establish/Baseline Sample Database for Integration Testing

- PURPOSE: Extract/construct/migrate the necessary data set from Legacy system.
- INPUTS:
    i. MRS
    ii. Migration Implementation Plan
- OUTPUTS: Populated/Validated/Baselined Sample Database
- TOOLS / METHODS:
    i. Data Transformation Tool/Custom 3rd party program (if required)
    ii. Software Engineering (Software Verification and Validation)
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Protection of sensitive (personal) data that may be used maliciously is vital. Verification of extracted or constructed sample data structure and content versus real data and planned data schema should be performed.

c. Perform Integration Testing

- PURPOSE: Deploy, integrate and verify the codes and sample data as planned in MIP
- INPUTS:

i. Migration Implementation Plan

ii. Sample database

iii. Adaptors

iv. Integration test procedures

v. MRS

- OUTPUTS: Test Results documented on Review /Test Report Form
- TOOLS / METHODS: Software Engineering (Software Verification and Validation)
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. Integration testing of interactions among converted programs, interactions between target and legacy systems and integration of programs with database(s) shall be considered together to maintain integrity.

ii. Additional tests (i.e. stress test) and evaluations should be applied together with formal tests. All problems should be recorded.

iii. Integration process shall be performed under CM.

iv. The solution shall be ready for acceptance testing phase as of completion of the all-open issues determined during integration.

d. Create Acceptance Test Procedures

- PURPOSE: Prepare the test procedure by improving the integration test procedure
- INPUTS:

i. Integration Test Procedure

ii. Review / test reports

- OUTPUTS: Acceptance Test Procedure with acceptance test sample data.
- TOOLS / METHODS:
    i. Software Engineering (Software Verification and Validation)
    ii. Peer Review
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Acceptance test procedure is prepared mainly by improving the integration test procedures. Acceptance test procedures are subject to peer review and formal approval.

6. Integration

The purpose of Integration is to verify and validate the converted programs in test environment and their interactions with legacy system and database.

This activity is further subdivided to the following sub activities:

a. Perform End User Training

b. Perform Acceptance Testing

Below each of these sub activities are further described:

a. Perform End User Training

- PURPOSE: Establish end user familiarity with the proposed solution.
- INPUTS: Updated Training Content
- OUTPUTS: Training Tracking Form, Training Evaluation Form
- TOOLS / METHODS: N/A
- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Information received from the training evaluation forms shall be used to update the training materials and end-user manuals.

b. Perform Acceptance Testing

- PURPOSE: Obtain formal approval prior to deployment.
- INPUTS: Acceptance Test Procedures
- OUTPUTS:
    i. Acceptance Test Results documented on review/test reports
    ii. Target Solution and Target Solution Documentation
- TOOLS / METHODS:
    i. Software Engineering
    ii. Peer Review
- ACTIVITY DESCRIPTION AND CONSIDERATIONS:
    i. Prior to the testing necessary physical and functional audits shall be completed as a predecessor activity.
    ii. Acceptance testing shall be conducted on actual target system site together with the formal participation.
    iii. Formal approval is necessary for each test case completed successfully.
    iv. All problems shall be recorded on Review/test report forms. Final approval shall be officially received from customer prior to deployment.
    v. Based on the review/test reports, necessary actions are taken for the revising the final Target Solution and Solution Documentation.

5.5. Deployment Phase

Table 12 Deployment Phase Overview

MAIN PURPOSE OF THE PHASE: The Deployment Phase is concerned with the cutover from the legacy system to the target system. When dealing with legacy systems, this process must cause as little disruption to the business as possible. The deployment phase results in the orderly transition of the enterprise from dependence on the existing legacy system to usage of the newly migrated system. This is not necessarily an instantaneous transfer, but it is a staged rollout of the new application, a gradual replacement of the current legacy system functions. The deployment phase starts with the DEPLOYMENT FREEZE operation and is completed with the CUT-OVER operation.
MAIN SEQUENCE OF ACTIVITIES OF THE PHASE: 1. Deployment to Production Environment in accordance with Deployment Plan 2. Perform Post Deployment Check 3. CUT-OVER TO THE TARGET ENVIRONMENT
MAIN INPUTS OF THE PHASE 1. Baselined Target System Code in Test Environment
MAIN OUTPUTS OF THE PHASE 1. Validated Target System for Cut-Over and COMPLETION OF CUT_OVER MILESTONE
MAIN TOOLS / METHODS OF THE PHASE 1. Target System and target database features. 2. Software Engineering and project management
KEY CONSIDERATIONS OF THE PHASE In accordance with the deployment plan, first task within the activity is the **"Deployment Freeze"** in order for the deployment to the production environment. Depending on the target system architecture this can be performed in varying forms / methods.

5.5.1. Main Activities

As it is listed in Table 12, the deployment phase consists of two main activities. These activities are described in further detail below:

1. **Deployment to Production Environment in accordance with Deployment Plan**

   - PURPOSE: Perform orderly transition of the newly migrated system in accordance with the deployment plan.
   - INPUTS:
     i. Base lined Target System Code in Test Environment
     ii. Migration Implementation
   - OUTPUTS: Target System Code deployed to production environment and connected to the production database
   - TOOLS / METHODS: Software and System Engineering
   - ACTIVITY DESCRIPTION AND CONSIDERATIONS:
     i. Prior to deployment an agreement shall be reached formally on a general “rollback strategy”.
     ii. Prior to deployment approval and acceptance of work done so far shall be validated.
     iii. Also, availability of adequate resources as stated in the deployment plan shall be validated.
     iv. Finally, all affected people shall be contacted about when changes will affect them.
     v. During deployment, all other stakeholders also need to be kept informed of project progress toward deployment.

2. **Perform Post Deployment Check**

   - PURPOSE: Ensure that the solution is working correctly following the deployment
   - INPUTS: Target System Code deployed to production environment and connected to the production database
   - OUTPUTS: Validated Target System Ready for Cut-Over
   - TOOLS / METHODS: Software and System Engineering
   - ACTIVITY DESCRIPTION AND CONSIDERATIONS:

     i. Validate that the solution is up, check security, Use control report to validate data integrity

     ii. User representatives shall be ready to run online and batch processes to make sure everything is working as expected in the target environment.

     iii. Check on interface systems and databases.

     iv. Many applications are integrated tightly, and a change to one may have an unexpected impact on another.

Communicate with stakeholders the status of the deployment.

3. **"CUT-OVER TO THE TARGET ENVIRONMENT"** is performed and the team gets ready for a formal hand-over for the solution to the operations team.

5.6. Operational Support Phase

Table 13 Operational Support Phase Overview

MAIN PURPOSE OF THE PHASE: The purpose of the operational support is to monitor the deployed solution for some period of time to ensure that the solution is stable and make the formal hand-over to operations.
MAIN SEQUENCE OF ACTIVITIES OF THE PHASE: 1. Monitor Deployed solution for stabilization 2. Hand-over to operations
MAIN INPUTS OF THE PHASE Target System with the related documentation
MAIN OUTPUTS OF THE PHASE Target System with the related documentation.
MAIN TOOLS / METHODS OF THE PHASE 1. Formal Review, Peer Reviews 2. Project Management
KEY CONSIDERATIONS OF THE PHASE N/A

5.6.1. Main Activities

As it is listed in Table 12, the operational support phase consists of two main activities. These activities are described in further detail below:

1. **Monitor deployed solution for stabilization**

- PURPOSE: Migration project team and operations work towards a predefined set of completion for the solution to ensure that the solution is stable.

- INPUTS:

    i. Problem/errors countered.

- OUTPUTS: Recording and correction of encountered problems.

- TOOLS / METHODS: Software and systems engineering.

- ACTIVITY DESCRIPTION AND CONSIDERATIONS: Depending on the contractual project formation, the monitoring and correction of problems phase can be continued for a fixed duration or for a full production cycle. The monitoring time is also dependent on the time period needed to run all of its functions.

2. Hand-over to Operations

- PURPOSE: Transferring the ownership and responsibility for the system to the operations team/organization.

- INPUTS:

    i. Target System

    ii. Known errors/outstanding issues

    iii. Enhancements requested.

    iv. List of stakeholder contacts.

    v. List of migration project team contacts.

    vi. Solution Manuals/Technical documentation

- OUTPUTS: Formal Hand-over letter.

- TOOLS / METHODS: Formal Meeting

- ACTIVITY DESCRIPTION AND CONSIDERATIONS:

i. Ensure a smooth transition by making sure that the employees of the operations and maintenance organization is involved to some degree in the project during the warranty period. This will allow them to have a basic understanding of the solution as well as some history of the project so they have some context for events that arise when the solution is moved to production.

Application Maintenance Manuals shall contain all of the technical information associated with the solution, including libraries and locations, database layouts, systems documentation, job schedule, interfaces, design diagrams, etc.

5.7. Advantages of the Framework

The advantages of the proposed framework can be summarized as follows:

- When compared to the other frameworks (on the industry or academy side) mentioned in the document, the proposed framework is specifically targeted for COBOL/CICS based mainframes. Due to this specialization, this increases the practical utilization of framework for these mainframe profiles.

- During the migration projects, it is necessary to use some tools (repository analysis, testing tools, etc) for many activities to increase the level of automation as much as possible. However, many frameworks; especially the ones from the vendors, are heavily dependent on their specific tools and they are intended to be used only with those specific tools. On the other hand, our proposed framework is not dependent on any specific vendor tool.

- The proposed methodology is provided with an easily understandable and systematic structure: All phases are presented with clear description of purpose, sequence of activities, main inputs, main outputs, main tools/methods and key considerations. Similarly, all activities within the phases are also presented with a clear description of purpose, inputs, outputs, tools/methods and activity descriptions.

- The migration is not only regarded as a technical exercise. Therefore, within the proposed methodology, both technical and administrative/managerial aspects of the migration project is taking into account.

- As it is not dependent on any specific vendor tool or approach and as it is designed with a standardized structure, it is open for growth and customization. Therefore, it is possible to do these further improvements or customizations either in specific phases or activities within those phases.

CHAPTER 6

CONCLUSION

6.1 Thesis Contributions

Within this thesis study, mainly two fundamental contributions are provided:

- The first one is the decision method proposed for the selection of the appropriate modernization strategy. A detailed comparative analysis of the modernization strategies is also presented within the decision method. The problem is clearly stated by [15] that *"there is no adequate decision method to help organizations decide on the investment strategy for the legacy systems"* and that the managers tend to make decisions based largely on instinct. Therefore we consider that this method can be used to overcome the stated problem for informed and systematic modernization strategy selection. To demonstrate and validate the usability and suitability of the decision method, it was applied with three different mainframe hosting organizations. The applied experimentation results for these organizations are also presented in the thesis study.

- The second one work and contribution provided is the proposed migration management framework for use in the delivery of migration projects. It is not dependent (or biased) on any specific vendor, technology or migration toolkit for the whole lifecycle of a typical migration project. The uniqueness of the methodology also comes from

the fact that it is specifically targeting the COBOL/CICS based mainframes. Among many different mainframe profiles, COBOL/CICS based mainframes are dominantly the largest population within the mainframe world. In order to avoid being too generic or impractical in the application, this specialization was chosen to provide a practical and applicable methodology.

Within this thesis study, a two staged approach was selected: Firstly the problem with the selection of the appropriate modernization strategy is handled. Before selecting the right approach for migration, first of all whether the migration is the right modernization strategy is to be determined. For each organization, depending on the various technical and non-technical parameters, the appropriate strategy may be different among migration, packaged application, rewrite and do-nothing alternatives. Secondly, for the case with selection of migration alternative; a migration management framework is proposed to handle the migration project.

The followings are the main limitations in the thesis study:

- Within the migration methodology, the support processes required during the delivery of the project are referenced only to the extent that there is a specific difference regarding migration from the standard IT projects.

- As there are many different forms of data and databases within the mainframes, data and database migration is a special and diverse branch of migration. Therefore, it is dealt only within the general frame of migration within the framework.

- As there are various migration tools available in the industry, a specific tool has not been developed as part of this thesis study. However, the migration tools to assist migration projects were classified and explained up to a certain detail in the thesis study. Moreover, the specific tools were also referred within the lifecycle stages of the methodology when needed. Similarly, a through analysis of these vendor tools is outside the scope of the thesis study.

6.2 Future Work

The future work of this thesis can be continued in the following directions:

- The proposed migration methodology can be applied within various large-scale migration projects and the experience gained with those projects can be shared with the researchers. This expertise can be used to improve the proposed methodology.

- The decision method proposed can be further extended in all those three dimensions.

- The integration of supporting processes (test engineering, reverse engineering, etc) within the methodology can be further studied for further specialization on those fields within the migration domain.

- The methodology can be transformed for mainframe profiles other than the COBOL/CICS mainframes.

REFERENCES

[1] The Gartner EXP CIO report, "Meeting the Challenge: The 2009 CIO Agenda", January 2009.

[2] International Data Corporation Report, IDC, 2003.

[3] Lei Wu, "Migrating Legacy System towards Object technology", *M.S. Thesis,* Universite de Montreal August 2005.

[4] H.Yang, M. Ward, "Successful Evolution of Software Systems", *Artech House*, ISBN 158-053349-3,2003.

[5] Magnus Ramage, Carole BrookeGold, Malcolm Munro, Keith Bennett, Nicolas Gold, "Overcoming The Legacy Dilemma", *Proceedings of Systems Modeling for Business Process Improvement, Coleraine,* 29-31 March 1999.

[6] Frederick P. Brooks, 1975 – "The Mythical Man-Month," *Addison-Wesley Publishing Co., Reading, MA*.

[7] M. Brodie, and M. Stonebraker, "Migrating Legacy Systems: Gateways, Interfaces and the Incremental Approach", *Morgan Kaufmann Publishers*, 1995.

[8] M. Brodie and M. Stonebraker, "DARWIN: On the Incremental Migration of Legacy Information Systems", *Technical Report TR-022-10-92-165 GTE Labs Inc*., March 1993.

[9] Gary Barnett, Mike Gilbert, "Legacy Renewal Strategies" 2006, www.ovum.com, last visited on December 2009.

[10] John K. Bergey, Linda M. Nortrop, Dennis B. Smith, "Enterprise Framework for the Disciplined Evolution of Legacy Systems", *Technical*

Report (CMU-SEI-97-TR-007, ESC-TR-97-007), Software Engineering Institute, October 1997.

[11] Flavien Salsmann, "Mainframes in IT Environments", *End of Study Thesis, SUPINFO International University, International Institute of Information Technology, July 2009.*

[12] The University of Edinburg and Heriot-Watt University, "System re-engineering patterns", *http://www.reengineering.ed.ac.uk/poster.html*, last visited on December 2009.

[13] Ben Wilson, Anubex Migration Center, "Chickens and turkeys migrate, but not necessarily in IT", *http://www.anubex.com/migrationscentre!chickensandturkeys.asp*, last visited on December 2009.

[14] NASCIO 2008 Survey of the States, "Digital States at Risk! Modernizing Legacy Systems", December 2008.

[15] K.H.Bennett, M.Ramage and M.Munro, "Decision Model for legacy systems", IEEE proceedings IEE Proc.-Softw., Vol 146, No. 3, pg 153 -159, June 1999.

[16] H.Nelson Weiderman, John K. Bergey, Dennis B. Smith, Scott R.Tilley, "Approaches to Legacy System Evolution", *Technical Report (CMU-SEI-97-TR-014), Software Engineering Institute*, October 1997.

[17] Santiago Cornella-Dorda, Kurt Wallnau, Robert C.Seacord, John Robert, "A Survey of Legacy System Modernization Approaches", *Technical Report (CMU-SEI-2000-TN-003), Software Engineering Institute*, April 2000.

[18] AbdelKarim Erradi, Sriram Anand, Naveen Kulkarni, "Evaluation of Strategies for Integrating Legacy Applications as Services in a Service Oriented Architecture", IEEE International Conference on Services Computing (SCC'06), May 2006.

[19] Bing Wu, Deirdre Lawless, Jesus Bisbal, Jane Grimson, Vincent Wade, D.O' Sullivan, Ray Richardson, "Legacy Systems Migration – A method and

its tool-kit framework" *Proc. of 3rd IEEE Conf. On Engineering of Complex Computer Systems*, 1997.

[20] Aversano L, Canfora G, De Lucia A., "Migrating legacy system to the Web: A business process reengineering oriented approach", *Advances in Software Maintenance Management: Technologies and Solutions, Polo M (Ed.). Idea Group Publishing: U.S.A., 2003; 151–181.*

[21] HM Sneed, "Risks involved in reengineering projects", *Proceedings of the 6th IEEE Working Conference on Reverse Engineering, Atlanta, GA, 1999. IEEE CS Press: Silver Spring, MD, 1999; 204–211.*

[22] G. Canfora, A. Cimitile, A De Lucia, GA Di Lucca, "Decomposing legacy programs: A first step towards migrating to client–server platforms", *Journal of Systems and Software 2000; 54:99–110.*

[23] G. Canfora, A. De Lucia, GA Di Lucca, "An incremental object oriented migration strategy for RPG legacy systems", *International Journal of Software Engineering and Knowledge Engineering 1999; 9(1):5–25.*

[24] M. Rahgozar, F. Oroumchian, "An effective strategy for legacy systems evolution", *Journal of Software Maintenance: Research and Practice 2003; 15:325–344.*

[25] B. Wu, D. Lawless, J. Bisbal, V. Wade, J. Grimson, R. Richardson, D. O'Sullivan, " The butterfly methodology: A gateway-free approach for migrating legacy information systems", *Proceedings of 3rd IEEE International Conference on Engineering of Complex Computer Systems, Como, Italy, 1997. IEEE CS Press: Silver Spring, MD, 1997; 200–205.*

[26] Andrea De Lucia, Rita Francese, Guiseppe Scanniello and Genoveffa Tortora, "Developing legacy system migration methods and tools for technology transfer", *Software-Practice and Experience, 2008; 38:1333-1364,* February 2008.

[27] HM. Sneed, "Encapsulating legacy software for use in client/server systems." *Proceedings of Working Conference on Reverse Engineering, Monterey, CA, 1996. IEEE CS Press: Silver Spring, MD, 1996; 104–119,* 1996.

[28] JG Butler, “Mainframe to Client/Server Migration”, *Computer Technology Research Corp.: Charleston, SC*, 1996.

[29] HM Sneed, E. Nyary, “Downsizing large application programs”, *Journal of Software Maintenance: Research and Practice,* 1994; 6(5):105–116.

[30] Chiang CC, “Wrapping legacy systems for use in heterogeneous computing environments”, *Information and Software Technology 2001; 43(8):497–507.*

[31] Lin JM, Hong ZW, Fang GM, Jiau HC, Chu WC, “Reengineering windows software applications into reusable CORBA objects”, *Information and Software Technology 2004; 46(6):403–413.*

[32] 20. Serrano MA, Carver DL, Montes de Oca C., “Reengineering legacy systems for distributed environments”, *Journal of Systems and Software 2002; 64(1):37–55.*

[33] Semih Cetin, N. Ilker Altintas, Halit Oguztuzun, Ali H. Dogru, Ozgur Tufekci, Selma Suloglu, “Legacy Migration to Service-Oriented Computing with Mashups”, *International Conference on Software Engineering Advances(ICSEA 2007), IEEE CS.*

[34] Bovenzi D, Canfora G, Fasolino AR, “Enabling legacy system accessibility by Web heterogeneous clients”, *Proceedings of the 7th European Conference on Software Maintenance and Reengineering, Victoria, BC, Canada, 2003. IEEE CS. Press: Silver Spring, MD, 2003; 73–81*

[35] Lewis G, Morris E, Smith D., “Analyzing the reuse potential of migrating legacy components to a service-oriented architecture”, *Proceedings of the Conference on Software Maintenance and Reengineering, 2006. IEEE CS Press: Silver Spring, MD, 2006; 15–23.*

[36] Grace A. Lewis, Edwin J. Morris, Dennis Smith, Soumya Simanta, “SMART: Analyzing the Reuse potential of legacy components in a Service-Oriented Architecture Environment”, *Technical Note*, *Software Engineering Institute (SEI) Technical Note (CMU/SEI-2005-TN-029)*, 2005.

[37] Dennis Smith, Liam O’Brien, Kostas Kontogiannis, “Working Session: Program Comprehension and Migration Strategies for Web Service and

Service-Oriented Architectures", *Proceedings of the 14th IEEE International Conference on Program Comprehension (ICPC'06) IEEE CS*, 2006.

[38] Stroulia E, El-Ramly M, Iglinski P, Sorenson P., "User interface reverse engineering in support of interface migration to the Web", *Automated Software Engineering 2003; 3(10):271–301.*

[39] Moore M., "User interface reengineering", *PhD Dissertation, College of Computing, Georgia Institute of Technology, Atlanta, GA*, 1998.

[40] Moore M, Moshkina L., "Migrating legacy user interfaces to the internet: Shifting dialogue initiative", *Proceedings of Working Conference on Reverse Engineering, Brisbane, Australia, 2000. IEEE CS Press: Silver Spring, MD,* 2000; *52–58.*

[41] Zhang Q, Chen R, Zou Y., "Reengineering user interfaces of E-commerce application using business processes", *22nd International Conference on Software Maintenance, 2006. IEEE CS Press: Silver Spring, MD, 2006; 428–43.*

[42] Jianjun Pu, Zhuopeng Zhang, Yang Zou, Hongji Yang, *"Reusing Legacy COBOL Code with UML Collaboration Diagrams via a Wide Spectrum Language", IEEE 2005.*

[43] A. De Lucia, G.A. Di Lucca, G. Canfora, A R. Fasolino, P. Guerra, S.Petruzzelli, "Migrating legacy systems towards Object-Oriented platforms", *Proceedings of IEEE International Conference on software Maintenance, Bari, Italy, Sept 1997. IEEE Comp. Soc. Press, pp 122-129*

[44] A. De Lucia, G.A. Di Lucca, G. Canfora, A. Citimile, "Decomposing Legacy Programs: A First Step Towards Migrating to Client-Server Programs", *The Journal of Systems and Software, vol. 54, 2000, pp. 99-100.*

[45] Microfocus Modernization Tools, www.microfocus.com, last visited on December 2009.

[46] BluePhoenix Solutions, www.bphx.com, last visited on December 2009.

[47] Andrey A.Terekhov, “Automating Language Conversion: A Case Study”, St. *Petersburg State University, LANIT-TERCOM, available at http://users.tepkom.ru/ddt/alc.pdf*, December 2009.

[48] Fujitsu Software, www.netcobol.com, last visited on December 2009.

[49] Dale Vecchio, “Can Software Clones Enable Mainframe Migrations”, *Gartner Report ID: G00134326, January 2006*.

[50] ATERAS, http://www.ateras.com/Home.aspx, last visited on December 2009.

[51] Metaware, http://www.metaware.fr/en/, last visited on December 2009.

[52] Dr. Mike Gilbert, Legacy Directions, Custom Research Note, Microsoft, February 2008.

[53] Gartner Report, IT modernization isn’t a choice, it is imperative”, *Report ID: G00155029*, September 2008.

[54] Metagroup, “Business Drivers Prompting Modernization Of Legacy Systems”, *Meta Group Survey*, 2004.

[55] Tamara Petroff, “Strategies for Legacy Migration Success”, *eAI Journal*, June 2002.