AN EVALUATION OF THE DEVELOPMENT OF A CASH MANAGEMENT SYSTEM

By

Monisha Naidoo (951019642)

Submitted in partial fulfillment of the requirements for the degree of

MASTERS IN BUSINESS ADMINISTRATION

Graduate School of Business, Faculty of Management
University of Natal (Durban)

Supervisor: Adv. Lee Gibson

June 2003

CONFIDENTIALITY CLAUSE

June 2003

TO WHOM IT MAY CONCERN

RE: CONFIDENTIALITY CLAUSE

Due to the strategic importance of this research it would be appreciated if the contents remain confidential and not be circulated for a period of five years.

Sincerely

M.Naidoo

DECLARATION

This research has not been previously accepted for any degree and is not being currently submitted in candidature for any degree.

Signed: .....

Date: .....

ACKNOWLEDGEMENTS

My special gratitude goes to:

- My parents and family for always supporting me in whatever I do. Nothing pleases me more than making them proud of me and I know by completing the MBA, it would do just that.
- My boyfriend, Ricky, who has shown love, support and understanding during this stressful but worthwhile time. He had also taken the time to assist me in acquiring research for this dissertation.
- The project team who assisted me in every way possible to get all the facts about the project. They even took the time to read through my dissertation to ensure that the information provided was accurate.
- My friends and work colleagues for all their assistance, motivation and encouragement throughout my academic pursuit.
- My supervisor, Adv. Lee Gibson, for all his invaluable time, guidance and encouragement throughout this phase.

ABSTRACT

Information systems and information technologies are the fastest growing industries in developed and developing countries. However, "...studies repeatedly point out that 30 to 45% of systems projects fail prior to completion. Over half of all systems projects overrun their budget and schedules by up to 200% or more" (Lientz & Rea, 1999).

The objectives of the research was:

- To determine best practices for developing an information system. Regardless of what information system it is, there are certain practices that will be applicable to all information systems.
- To evaluate the development of an information system in a financial institution, and
- To establish ways to improve the development of information systems.

The study was conducted amongst the project team that developed the information system and the users of the system. The findings of the study indicated that problems were experienced during all phases of the SDLC. It was evident that the incorrect procedures in the initial phases of the SDLC, caused problems throughout the entire development process.

Implementing the recommendations proposed would enable the project team to successfully implement an information system that meets the user requirements.

TABLE OF CONTENTS

LIST OF FIGURES

PAGE

LIST OF TABLES

PAGE

CHAPTER ONE: INTRODUCTION

1.1 Introduction

Information systems (IS) and information technologies are the fastest growing industries in developed and developing countries. Huge amounts of money continue to be invested in these industries. Due to pressure of time-to-market, there is a corresponding pressure to increase productivity. To maintain a competitive edge in today's fast-changing world, an organisation's success depends on effectively developing and adopting IS. Despite significant efforts to improve systems project success, many still fail (Whitten, Bentley & Barlow, 1994 p.10).

"Studies repeatedly point out that 30 to 45% of systems projects fail prior to completion. Over half of all systems projects overrun their budget and schedules by up to 200% or more" (Lientz & Rea, 1999). Current literature indicates that most of the system project problems are related to management, organisational, human, and cultural issues – not technical problems.

1.2 Background of the study

Due to the sensitivity of information in this study, the name of the financial institution and the system being evaluated will not be mentioned.

The financial institution's Information Technology (IT) Department developed a system that was to replace the existing IT cash management system that manages the notes and coins in the vaults of the seven branches. The primary goal of the project is to enable the management of bulk cash on a national level by redesigning, rewriting, enhancing and integrating various legacy systems and by providing additional functionality to interact with external parties. The system has been designed to use the web so that branches and external parties can access the system.

The project consisted of a number of phases and iterations within the phases. The first phase of the system, which was to provide for the full functionality of the current cash management system, has been completed, but the project has not yet been signed off due to a number of reasons that could have been avoided. Many of the project milestones had to be rescheduled, some of which were missed.

The focus of this research, therefore, was to determine best practices for developing such a system from the planning phase to the implementation phase. The research will also point out reasons why systems development projects fail and how these issues can be avoided.

1.3 Motivation for the research

The motivation for the research is that many system development projects fail and companies prefer not to document reasons for failure due organisational politics. Therefore, many development projects tend to fail for similar reasons. By doing this research, it was envisaged that development teams learn from the mistakes and benefit from the recommendations should they be facing a similar situation.

1.4 Value of the project

This research will be invaluable to the project team that designed the system as there are still more versions that will be released. Before analysis starts for the next version, this research will provide valuable insight of all the aspects of the systems development life cycle that were performed well and those that were not. By highlighting the problems experienced, the project management team could be more aware of these issues and could make a more assertive effort to prevent the same mistakes from happening again.

The research also provides recommendations that could be implemented in the next version to avoid the same mistakes from reoccurring. This research will explain most, if not all, the reasons why the project experienced the problems it did.

1.5 Problem statement

Why did the project undertaken by the financial institution's IT Department experience several problems that caused the project to run over schedule?

1.6 Objectives of the study

The objectives of the research was:

- To determine best practices for developing an information system. Regardless of what information system it is, there are certain practices that will be applicable to all information systems.
- To evaluate the development of the information system created by the IT department of the financial institution, and
- To establish ways to improve the development of information systems in the IT department.

1.7 Research methodology

The research methodology is qualitative of nature. In-depth interviewing was conducted with the project manager, project leaders, business analyst, database administrator, developers of the project team and end users of the system.

Secondary data was used to obtain more insight about the development of information systems. Documentation prepared for the project and other projects developed by the IT Department in the financial institution was researched. A literature search of books, journals and the World Wide Web was conducted on the subject of developing an information system.

1.8 Limitations of the project

An information systems development project encounters many challenges like business issues, human issues and technical issues.

Business issues include, for example, when the IT project team changes, the project team does not consist of qualified people, disagreements within the project team which cannot be consistently resolved, lack of commitment from the project team, etc. Human issues include the users and their commitment to the project. This will also include problems experienced with user requirements as well as user resistance to change. Technical issues include, amongst others, the development tools and the maintenance of the various versions, the performance of technology is not satisfactory, etc. (Lientz & Rea, 1999).

This research focused on the human issues affecting the development of an information systems project. Business issues was not researched in this study as these issues are faced by all businesses and the list could be endless. Each information system development project is faced with internal politics which needs to be resolved prior to commencement of the project. The business issues also need to be monitored throughout the project to ensure that they do not affect the performance of the team.

1.9 Structure of the study

The summary of each chapter is as follows:

Chapter Two: This chapter consists of an extensive literature review and is aimed at developing a best practice for developing an information system.

Chapter Three: A detailed review of the information system developed in the financial institution's IT Department. This chapter describes the procedures followed by the project team to develop the information system.

Chapter Four: This chapter focuses on evaluating what was performed well and what was not in the project based on best practices determined in chapter two.

Chapter Five: The findings of the study are established and thereafter recommendations and conclusions are made taking into account the review of the former chapter and the analysis of the latter chapters.

1.10 Summary

The central focus of the study was to evaluate the development of the cash management system developed by the financial institution's IT Department. The systems development life cycle (SDLC) is a framework for information systems development. The project was evaluated according to each phase of the SDLC. In order to do this, best practices for developing an information system should be established.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

To maintain a competitive edge in today's fast-changing world, an organisation's success depends on effectively developing and adopting information systems. However stories about delays, cost overruns, and abandonment of software projects are widely reported in the literature. In other industries, causes of project failures are investigated and reports written, but in the computer industry their causes are covered up or ignored. As a result, the IT/IS industry keeps making the same mistakes over and over again (Hartman & Ashrafi, 2000 pp.5-6).

"A systems development life cycle is a systematic and orderly approach to solving business problems, developing and supporting resulting information systems" (Whitten, Bentley & Barlow, 1994 p.11).

Systems development is not a 'hit-or-miss' process. As with any product, information systems must be carefully developed. Successful systems development is governed by some fundamental, underlying principles. Systems development life cycle is a disciplined approach to developing systems. Although such an approach will not guarantee success, it will improve the chances of success. Most experts agree that there is a life cycle, but beyond that, there's little agreement. There are many versions of the SDLC and although their terminology differs, they are more often alike than different (ibid. p.13).

The SDLC consists of five phases, namely, Systems Planning, Systems Analysis, Systems Design, Systems Implementation and Systems Support. The phases of the project should be completed in sequence, however, at any given time, one may be performing tasks in more than one phase simultaneously. Furthermore, one may have to backtrack to previous phases and activities to make corrections or to respond to new requirements. Obviously, one needs to be very careful about backtracking, as this may lead to never implementing the new system (ibid. p. 92).

2.2 Planning

"Planning is the ongoing study of a problem environment to identify problem-solving possibilities" (ibid. p.11). Ideally, the projects that are selected will provide the greatest long-term benefit to the business. Thus, planning of information systems cannot be separated from the planning of the business itself.

2.2.1 Project management

"Current literature indicates that most of the software project problems are related to management, organisational, human, and cultural issues – not technical problems" (Hartman & Ashrafi, 2000 p.5).

IT projects continue to run over time and over budget, resulting in systems that do not match business or end-user requirements, or stall before they are finished. The only possible fault one can call common to all failures is inadequate management, to a greater or lesser extent, of those projects. This in turn means that blame must be laid on the shoulders of the project manager responsible. In mitigation, the design and implementation of IT systems are complicated, involving complex matrix of technologies and business inter-dependencies, which are shifting at different rates across several management planes (Bocij, Chaffey, Greasley & Hickie, 1999 p.308).

Good project management is about taking account of change up-front, building in risk management and contingency planning buffers. "It is about setting realistic duration and cost estimates, and not being afraid to tell the chief executive that his pet project cannot be finished in the six months he expects, even if being so honest puts your job at risk" (ibid.).

a) Establishment of standards for consistent development and documentation

i. Use of unified tools

Effective development depends on the effective use of software development tools. However, unless the project managers choose the right tools and train their teams to use them correctly, the tools may absorb more time and attention than the processes they are supposed to support. Many teams waste precious time and resources trying to use unsuitable tools that are forced upon them by organisational policies. Wars over methodology, languages and tools will spell disaster (Cardozo, 2002 p.3).

Also, huge projects usually involve team effort. Projects can get derailed if members are not using the same toolset or using it according to project guidelines. When a project member becomes ill and no one knows where his or her code resides on the network, or a new team member comes on board that is unfamiliar with the practices of other developers on the team, the project is at risk. Every developer brings his or her experience and habits to the project, and project managers should leverage this experience to improve performance – but only if it fits the project (ibid.).

ii. Project methodology

A major source of failure is the methodology itself. The problem with many approaches to developing projects is that it doesn't provide a complete picture of project progress until near the end of the project, at the system implementation phase, so one may not be able to detect serious problems until that point. This phenomenon, known as "late design breakage", can result in unnecessary rework and a lot of stress on people and budget. This happens because the initial architecture is based only on a part of the problem space. The result is that the system's architecture never stabilises, and the project team spends many hours doing unnecessary rework to the architecture instead of adding functionality (ibid.).

One of the pitfalls of iterative development is that one can get oneself into a never-ending sequence of iterations. When this happens, the project will be late and run over the budget. In most cases the project team begins enthusiastically but then gets lost somewhere near the "end". A way to avoid this is to begin with the end in mind. Iterative projects should be structured around goals (ibid. p.5).

iii. Establishing Standards

Systems development standards usually describe activities, responsibilities, documentation guidelines or requirements and quality checks.

An organisation has many information systems that may include thousands of programs and software packages. If each analyst/programmer were to adopt their own preferred SDLC and use their own tools and techniques to develop and document systems, a state of chaos would result. In medium to large information systems, systems analysts and programmers (and users) come and go as they may be promoted or transferred or resign. "In order to promote good communication between this constantly changing base of users and information systems professionals one must develop standards to ensure consistent systems development" (Whitten, Bentley & Barlow, 1994 p.93).

"The need for documentation standards underscores a common failure of many analysts – the failure to document as an ongoing activity during the life cycle" (ibid. p.93). Most professionals tend to do post documentation. Documentation should be a working by-product of the entire systems development effort. Documentation reveals strengths and weaknesses of the system to others – before the system is built. It stimulates user involvement and reassures management about progress.

b) Attacking risks

Every project is confronted with risks that may include design flaws, ambiguous requirements, an inadequate development environment, and so forth. It is almost impossible to recognize them all on day one. At project start-up and near project closeout, most risks are associated with the project's environment: Do we really have users? Are the users trained and ready for deployment? When the project team starts to analyse requirements, however, risks are increasingly associated with technical issues. "The risk associated with a "big-bang" integration strategy is that the design may not reflect the (real) requirements. Late discovery of design defects can then cause budget and schedule over-runs, which may eventually kill the project" (Cardozo, 2002 p.4).

"During project initiation, most risks fall within the project environment, which includes the organisation, funding, people, tight schedules and expected benefits of new technology. Developing a business case is a crucial step" (ibid. p.7). A business case promotes understanding of the business problem and buy-in from the project sponsors. It also helps to explain the project's business drivers to other stakeholders. Furthermore, a business case is the most powerful weapon against feature creep. Developing one must be a joint effort by the project team – which is responsible for determining development costs and schedule - and the user (project owner), who is responsible for defining the benefits.

Adopting a proactive attitude also means that one must act when confronted with risks. For example (ibid. p.5):

- When the scope of the iteration turns out to be too large to deliver on schedule, reduce the scope and deliver a smaller solution.
- If the users are not able to visit the development site, try to go to them.
- If the users cannot seem to express their ideas about the user interface, then develop a prototype they can react to.
- When a project depends on services to be developed by another project, create stubs in case the other project does not deliver on schedule.

c) Developing the 'delivery habit'

The 'delivery habit' reflects a proactive attitude. When the team develops iteratively, project members work cooperatively on more artifacts within a tighter timeframe, and they do not sit around waiting for someone else to finish an activity before they get started. They realise that they can get a lot done, even if the other person is only halfway through (ibid. p.4).

With iterative development developing parts in sequence mitigates risks, so that the system evolves instead of being constructed and integrated all at once, near the end of the project. From a cultural perspective, this means that the project team must adopt a "delivery habit" that ensures progress (i.e. a demonstration-based approach). With each iteration, they will mitigate more risks and deliver more function to the user. Progress will be measured by the results of systems tests and user feedback that indicate which requirements are now specified, designed, incorporated, tested, or deployed. If delivery stops, there will be no visible progress, and the project will be in danger (ibid.).

d) Scope management

The return on investments (ROI) for software projects can be improved by reducing the project size. This can mean reducing the amount of code required for the product to fulfill the needs of the business, and/or reducing the number of system features. "In most systems, 20% of the features solve 80% of business needs" (ibid. p.9).

One way to manage scope effectively is to adopt a use-case-driven approach. This means that Use Cases are the basis for the entire development process. Traditional software development approaches use functions rather than Use Cases. Functions are not directly related to business value. Use Cases more or less tie system functions together. They describe what the system must do in the language of the customer, so they are understandable by a wide range of stakeholders. They also form the basis for estimating, planning, defining test cases and test procedures, and trace ability.

e) Project management methodology

"Project management methodology is built around the idea that a project is required to deliver a product(s) within the time, cost and quality constraints imposed" (Bocij et al, 1999 p.320). The products are defined not just in the sense of the technical product of the delivered IT system, but including management products such as project plans and quality products such as quality reviews.

i. Project Plans

There are three levels of plans, each of which consists of a technical plan (detailing which activities are required) and a resource plan (giving which resources are needed). The three levels of plans are (ibid. p.321):

1. *Project plan.* This shows the main activities within the project, providing an overall schedule and identifying resources needed for project implementation.
2. *Stage plan.* A stage plan is produced at the end of each previous stage in the project. The project board reviews all progress against the plan and takes corrective action as necessary.
3. *Detailed plan.* If a project is already broken down into stages, a detailed plan may not be required. However, for large projects with few stages, a series of detailed plans may be needed.

There are also two additional types of plan to complete the planning structure (ibid.):

4. *Individual work plan.* This provides the allocation of work of a project. This information is extracted from tasks listed in the stage plan or detailed plan.
5. *Exception plan.* Exception plans enable 'out-of-control' behaviour within a stage plan to be reported to the project board. This is required if the project moves outside tolerance margins set by the project board. The exception plan replaces the stage, detailed and individual work plan for that stage.

The project plan is created during the project initiation stage and provides an overall assessment of the cost, time and resources necessary to undertake the project. The stage, detailed and individual work plans are more detailed and provide a basis for day-to-day control of project activities. If the actions within the exception plan are accepted, it will replace the stage plan for the remainder of that stage.

ii. Project control

Project control is the activity of ensuring that a project meets planned objectives.

Business integrity involves ensuring that the work is carried out to the schedule agreed within the resource and cost constraints imposed. *Technical integrity* involves ensuring that the development system meets the goals of quality, reliability and maintainability (ibid. p.324).

Control is exercised by comparing performance to plan and taking action on any deviation that is outside the agreed tolerance. Management tolerances measure deviation from planned cost or schedule, while technical tolerances measure deviation in quality as defined by the user requirements and objectives.

iii. Management control

Management controls are in the form of meetings of project staff that produce a set of predefined documents. These allow senior management to assess the status of the project before providing further expenditure.

- *Project initiation*

The outcome of this stage is a project initiation document that will include a high-level plan for the project and confirmation of the responsibilities of project members. There will also be a more detailed plan of the first stage of the project (ibid. p.325).

- *Stage assessment*

There are checkpoint meetings held on a regular time-related basis to review progress against plans, particularly in connection with individual work allocations. They are held at a team level and are usually run by the stage manager or team leader. Highlight reports provide a regular summary of progress to date to the project board. The end stage assessment is not time based but is triggered by the end of each project stage. The mid stage assessment is an optional event and may be triggered by the following (ibid. pp.325-326):

- A need to check programs during a length project stage,
- When stage tolerance levels have been exceeded, and
- When it is felt necessary to begin the next stage before the end stage assessment can be held for the present stage.

- *Exception plans*

If a stage cannot be completed within its tolerances, the project manager must advise the project board immediately and present an exception plan as a mid-stage assessment. An exception plan consists of the technical plan covering remaining stage activities, a matching resource plan and additional information to describe the exception. This should include the impact of options considered in the stage plan, project plan and business case. If the project board agrees with the exception plan, it becomes the stage plan for the remainder of that stage (ibid. p.326).

- *Project closure*

The project closure meeting replaces the final end-stage assessment and confirms the signing of the system, user, operations, security and business acceptance letters by the appropriate board members. The acceptance criteria should have been clearly stated in the project initiation document (ibid.).

- *Product controls*

Product controls ensure that quality is built into the development of the products during the project. Tasks involved include agreeing on quality criteria for products with users, planning quality reviews and detecting and correcting quality problems as early as possible. The measurement criteria for a product's quality are contained within the product description and as such are created during the planning stage, thus building in quality to the product design (ibid. p.327).

- *Quality reviews*

The quality review is to identify errors through a planned and documented process as early as possible in the development cycle. The quality review consists of three phases (ibid. p.327):

1. *Preparation*. This includes setting up a review team and distribution of appropriate documentation.
2. *Review*. The meeting is held and actions are listed and allocated to individuals.
3. *Follow*-up. This covers the correction of actions listed.

- *Configuration management*

Configuration management identifies each hardware, software or documentation component used and records the status of that component. Configuration management is needed because of the dependence between components within a project. Each component will have its own development cycle and during this development any of the components may be changed, which could make them incompatible with other components (ibid. p.328).

2.2.2 User involvement

"User involvement in the process of developing information systems has long been known to be a critical component of eventual success" (Jiang, Chen & Klein, 2002 p.20). Because formal requirement determinations occurs early in the systems development lifecycle, improvements are likely to influence the overall quality of systems development efforts significantly and, therefore, have the potential to reduce development costs dramatically (Browne & Ramesh, 2002 p.625). Lin & Shao (2000 p.292) also mention that getting users involved in the development process may also improve their attitudes toward the system and enhance the importance and relevance users perceive about the system.

Successful organisations have come to understand the need to design products or services to meet customer requirements and expectations. User-centered design has become a business strategy that many companies use to gain competitive advantage and maintain economic viability. Organisations realise that they cannot rely on designers, developers, or specialists to know how to design products and services to meet customers' needs. Since designers are seldom the primary users of their systems or products, their own biases, rationalizations, and views often interfere with assessing what customers truly need or want. The more customer contact a project has, the more likely it is to be successful (Smart & Whiting, 2001 p.177).

"To ensure that the individual projects actually meet the needs of the business users, it is vital to drive business knowledge into IT through requirements and business process modeling" (LeClair, 2002 p.3). There is no value to the business for systems that don't meet the user's functional requirements. This process must also address the user's service-level expectations. Response time and system availability metrics are just as important as features and functions in successful deployment.

a) Project stakeholders

A proactive attitude also helps in developing a partnership with project stakeholders and establishing effective communications. One needs to demonstrate that the business needs the project is designed to meet are understood and that the project team is committed to building the right solution for the right problem. One should also explain the development process to stakeholders and show how it supports building the right solution.

To be proactive, it is essential to determine *who* the stakeholders are: Who influences and who makes decisions? This activity is known as stakeholder analysis. Once these people are identified, one must think of ways to get them on board (or deliberately not bring them in).

In general, there are four types of stakeholders (Cardozo, 2002 p.5):

- End users: People who will use the product.``
- System users: People who will keep the product "alive" during its post-deployment lifecycle (i.e., maintenance and support personnel, suppliers).
- Temporary users: People who develop the product or are involved with the product rollout (for example, the project team, engineers, marketing people, trainers).
- Other Stakeholders: People who are not directly involved in the project but have the power to either make or break it (for example, management, laws and regulations, other projects, environmental movements).

Efforts should be made to involve as many staff as possible in the development. While it will not be practical to involve everyone, representatives of all job functions should be polled for their requirements for the system at the analysis stage. As many user and manager representatives as possible should be involved in the active analysis and design involved in prototyping.

Promotion of the system can also be achieved by appointing particular managers to champion the new system (Bocij et al, 1999 p.474):

- Senior managers or board members are used as system sponsors. Sponsors are keen that the system should work and will fire up staff with their enthusiasm and stress why introducing the system is important to the business and its workers.
- System owners are managers in the organisation who will use the system to create the business benefits envisaged.
- Stakeholders should be identified at every location in which the system will be used. These people should be respected by their co-workers and will again act as a source of enthusiasm for the system. The user representatives used in specification and testing can also fill this role.
- Legitimisers protect the norms and values of the system; they are experienced in their job and regarded as the experts by fellow workers; they may be initially resistant to change and therefore need to be involved early.
- Opinion leaders are people whom others watch to see whether they accept new ideas and changes. They usually have little formal power, but are regarded as good 'ideas' people who are receptive to change and again need to be involved early in the project.

There is also a critical need to identify and manage realistic expectations of the stakeholders to achieve perceived project success. This can only be done through effective communication.

b) User support

"Numerous studies have demonstrated the negative effects a lack of user support, including resistance to change and unwilling involvement, has on project performance. Approaches to mitigate the risk of low user support have long received a tremendous amount of attention from information systems project managers and researchers" (Jiang, Chen & Klein, 2002 p.20). As a result, various user-support risk controls have been tried to improve user involvement and participation, for example, the use of prototyping, project ownership, and requirements sign-offs.

These techniques have helped define system requirements when a lack of user support occurred. However, these techniques are *reactive* to a lack of user support during the systems development process. "If one wants a project to be successful, one needs to motivate the project owner to have users intimately involved in identifying requirements during the initial stage of the project" (Boyette, 2002 p.1). Explaining the benefits of user input into the project requirements helps the user to become more willing to spend the necessary time up front to create a finished product that is user-friendly and that meets the needs of the entire business.

To this end, many experts argue prevention of a lack of user support requires good communication and a positive relationship between users and IT staff before and during a project. "Effective communication and positive relationships must be cultivated and planned as any other successful component of project management" (Jiang, Chen & Klein, 2002 p.20). Preproject partnering refers to a philosophy in which stakeholders work together before the project begins. The intent behind preproject partnering is to build a foundation among stakeholders for collaboration. In addition to identifying key stakeholders and their objectives, partnering emphasizes the activities of identifying potential conflict areas, providing a process for conflict resolution, and incorporating a continuous improvement component into the project process.

c) Communication

This is one of the most common sources of failure on development projects. Communication is a prerequisite for effective coordination, as it is the vehicle through which personnel from multiple functional areas share information critical to the successful implementation of projects. A well-performed project start-up can lay a foundation for effective communication. The goal of a project start-up is to establish a credible basis for the project that is acceptable to all stakeholders.

Project managers too often fail to seek answers to some fundamental questions (Cardozo, 2002 p.2):

- What business objectives/benefits is the project intended to achieve?
- What level of quality is expected for the end products(s)?
- What risks did the customer consider in deciding to set up this project?

"Getting the answers to these questions requires effective communication with the customer; building a project around these answers requires effective communication among project team members" (ibid. p.3).

d) Conflict

Since none of this is rocket science, it begs the question as to why IT projects continue to fail. "It's because there's too often a lack of an agreed requirements specification. At least 50% of the time of the contract should be to find out precisely what users and departments require" (Bocij et al, 1999 p.308).

Conflicts arising from disagreement among stakeholders can adversely impact the project development process. These conflicts arise from the diversity of interests in the final product. The different views arising at different times can potentially throw the project off course. But a formal process of resolving conflict and incorporating a diversity of ideas established prior to the start of the project can minimize the threat to the process. Preproject partnering is directed at resolving these conflicts to impact the performance of the project development process (Jiang, Chen & Klein, 2002 p.21).

2.3 Analysis

"Analysis is the study of the problem environment and the subsequent definition and prioritization of the requirements for solving the problem" (Whitten, Bentley & Barlow, 1994 p.11). Throughout analysis, the emphasis is on the business, not the computer.

2.3.1 User participation

User participation has long been regarded as an important factor to improve the chances of the success in developing an information system. It refers to the various design related behaviours and activities that the target users or their representatives perform in the systems development process. Through participation, users of an information system can interact with system designers in the stages of planning, analysis, design, testing, and implementation and, hence, aid in many aspects of the system development process. A variety of development methodologies, such as co-development, participative design and joint application design (JAD) have been proposed to operationalise user participation (Lin & Shao, 2000 p.283).

There are a number of benefits which can be expected of such user participative behaviours. User participation in systems development can enhance system quality through a more accurate and complete identification of user information requirements, knowledge and expertise about the organisation the system is intended to support, avoidance of unacceptable or unimportant system features and a better understanding about the system. User participation is also believed to increase user acceptance about system capabilities, an opportunity for users and designers to resolve conflicts about design issues, user's feelings of ownership toward the system, a decrease in user resistance to possible changes incurred by the system and a greater commitment from users. In consequence, user participation has been extensively sought and encouraged by practitioners in developing IS (ibid. p.283).

"Analysts and programmers frequently refer to 'my system'. This attitude has created an 'us-versus-them' attitude between analysts/programmers and their users" (Whitten, Bentley & Barlow, 1994 p.91). Although programmers and analysts work hard to create technologically impressive solutions, those solutions often backfire because they don't address the real organisation problems or they introduce new organisation or technical problems. For this reason, user involvement is an absolute necessity for successful systems development. The individuals responsible for systems development must make time for users, insist on user participation and seek agreement from users on all decisions that may affect them.

Misunderstandings continue to be a significant problem in systems development. However, user involvement and education minimizes such misunderstandings and helps to win user acceptance of new ideas and change (ibid. p.92).

Cardozo (2002 p.2) also points out that customer buy-in on the development process is crucial. Insufficient end-user involvement is the number one reason why projects fail. At the beginning of the project-request phase, good user representation should be encouraged by having a user or team leader involved in requirements gathering. After getting a commitment from the client for user involvement, be sure that they provide their processes as well as requesting a process flow diagram, in addition to any training or quality-control documents they might have. Users will often point out omissions or details that will make an application unusable if not included.

The proposed process will help the users and the development team to envision "what-if" scenarios and identify interactions with other groups that might have implications for the project. Most of the time, unfortunately, these steps are not taken and users don't see the application until user acceptance testing. By that time, the application is already built and changes would delay delivery (Boyette, 2002 p.2).

2.3.2 Information requirements determination

The importance of doing a good job of identifying needs and specifying requirements cannot be overstated. "It is recognized a project's needs begin a series of events that ultimately result in the production of a deliverable design to satisfy the user's wants" (Jiang, Chen & Klein, 2002 p.20). Traditional project approaches focus on how to identify these wants. These approaches provide excellent guidelines to help understand the functions of the current system and the flows of information, determining available inputs and desired outputs, locating interested parties, and determining the nature of stakeholders' interests. The approaches are not designed to prevent a lack of user support during the entire development process. Since a careful needs analysis at the start of the project development process usually is incomplete and premature, project managers must involve users throughout the entire development cycle. It is therefore imperative that steps be taken to ensure cooperation from the inception of the project concept (ibid.).

Information requirements determination (IRD) is the most critical phase of information system development. "IRD is a set of activities used by a systems analyst when assessing the functionality required in a proposed system" (Browne & Ramesh, 2002 p.625). Types of information gathered include goals for the system, business processes, data needs, design constraints and behaviour of users. Such information is commonly sought from the eventual users of the system through interviews, surveys, or observation, or may be derived by studying the systems currently being used in the organisation. This assessment of user needs is one of the key determinants of the ultimate success of an information system. However, because understanding human and organisational needs is difficult and complex, requirements determination is, in general, ad hoc and poorly understood. Further, the large number of completed systems that do not meet user specifications and expectations suggests that the determination of such requirements can be improved (ibid.).

The requirements determination process during the analysis phase can be divided into three stages: information gathering, representation and verification.

In the information gathering stage, the analyst uses his or her prior experience and knowledge to gather information about the functional, non-functional, and technical requirements for the proposed system. This may be accomplished in several ways, for example, by asking and/or observing how people perform tasks that will be supported by the new system, by examining business documents and forms and by the analyst's use of inference and imagination to envision user needs. The outputs of this stage vary, but generally take the form of notes, outlines, checklists, and informal diagrams prepared by the analyst. This information is used as input to the second stage, in which different representational techniques may be used to document the elicited requirements. The outputs of this stage may include informal diagrams, semi-formal diagrams (for example, data flow diagrams) and prototypes. These representations are then typically used to help verify that the requirements elicited are in fact correct. Users then sign a requirements document and the diagrams representing the requirements are given to a systems designer (ibid. p.626).

Throughout the stages of IRD, feedback loops are included to signify the iterative nature of the process. At each stage, the analyst assesses the quality and completeness of the outputs, and, if necessary, repeats activities within or between stages of the process. When the analyst is satisfied with the requirements, he/she terminates the IRD process.

- **Problems in requirements determination**

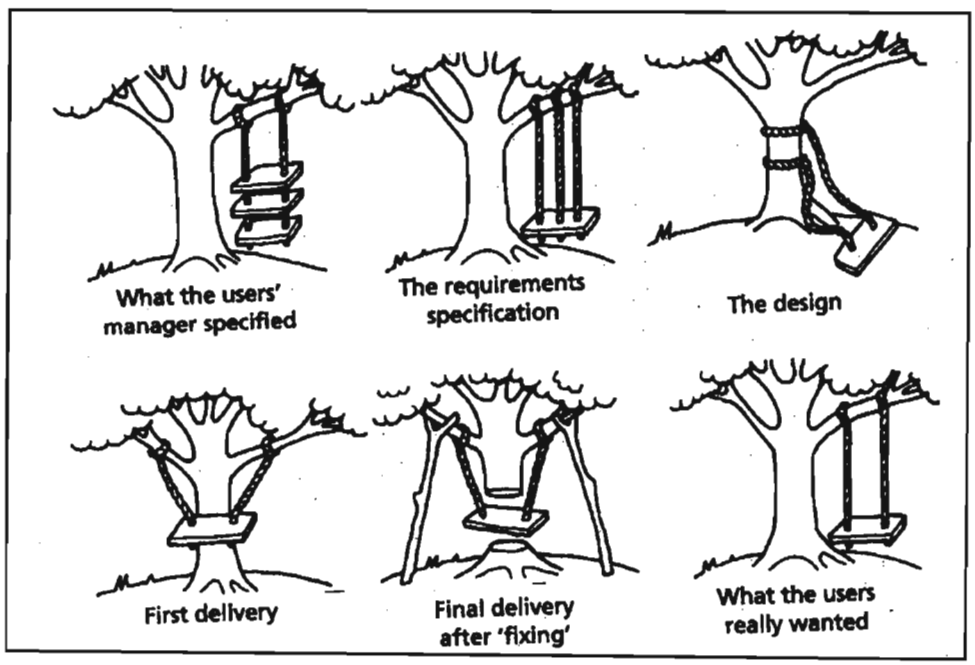
Despite the efforts of analysts and other parties involved in systems development, failures of requirements determination represent one of the leading causes of systems failure. There are four sources of difficulties (ibid. p.627): (1) constraints on humans as information processors; (2) the variety and complexity of information requirements; (3) communication issues between analysts and users; and (4) the unwillingness of users to provide requirements. These difficulties are basic to human problem solving and are independent of the systems development methodology being used.

- **Poor requirements gathering and documentation**

Another fail factor lies in the way requirements are treated and documented. In some projects, requirements engineers seem to focus mainly on the customer, disregarding other stakeholders. They do not recognize that requirements must be unambiguous for a whole range of stakeholders, including developers and testers. In additional, traditional software development approaches typically document requirements as functions, which are not directly related to business value. This makes them hard to prioritize, thus compromising scope management and project steering. Another problem with functions is that they make it hard to develop test cases. Test designers are forces to guess at scenarios that will cover the real usage of the system from the user's point of view, when in fact these should have been captured by the requirements (Cardozo, 2002 p.3).

Fig 1 shows how a user's requirements of a swing might be interpreted, not only at the requirements stage but also throughout the project.

Fig 1: Varying interpretations of a user's requirements



Source: Bocij et al, 1999 p.361

The figure highlights the importance of keeping all project stakeholders informed about the project goals to ensure that their expectations are aligned with the project plan.

- **Why are requirements gathering so much work?**

The next best thing to being there is to inspire the customers so that they are willing to be ones eyes and ears as they define user requirements.

Developing a web-based application or web site is even more detailed. In such a case, user requirements must answer dozens of questions (Boyette, 2002 p.3):

- What must the user do first, next and last for each function of the job?
- Do users enter comments after the function is completed?
- Does the application need to interface with another application?
- What are the user security requirements?
- Will the system have an administrator and department administrators, or will users self-register?
- Is there a need for system-stamped time and date and last-updated-by details?
- Will there be a need for reporting of the system usage or work results?
- Is every piece of data in the report tied back to the field source in the application? What are the computations?
- What are the business rules attached to each field on each page of the site?
- When "Enter" is selected, does one go right to a new function or to a list of pending items?
- Define field attributes: Length? Alpha or alphanumeric? Any validation required?

Describing someone's workflow in exact detail and in the proper sequence is difficult or sometimes impossible. One strategy for helping the design team is to have them shadow users of the new system as they do their work, gaining a much more accurate picture of what the new system must accomplish (ibid. p.4).

Success in this stage can mean a successful rollout, and though user requirements gathering will take some time, the end result is time saved, accurately defined needs therefore a smaller chance of scope creep, and a user-friendly application that meets the client's business needs and streamlines the users' jobs (ibid. p.5).

2.3.3 Analysis methodology

- **Fact finding**

The first task in analysis is to conduct a fact-finding exercise so that the information systems requirements can be determined. The methods that an organisation uses in the analysis phase will depend on two factors (Bocij et al, 1999 p.360):

- *Levels of decision-making involved:* A new information system will be under consideration either to resolve a problem or to create an opportunity. In either case, the objective is to improve the quality of information available to allow better decision-making. The type of system under consideration may include a transaction processing system, a management information system, a decision support system, a combination of these or some other categorisation of a system.

- *Scope of functional area:* A new information system may serve the needs of one functional business area or it may cut across many functional areas. An information system that is restricted in scope may be faced with fewer of the problems that can affect new systems designed to meet the needs of many different areas. As before, the techniques of fact-finding may be similar, but how they are used and the findings presented may be radically different. Organisational culture, structure and decision-making processes will all have a part to play in selling the systems solution to all the affected parties.

Regardless of the scope and organisation levels involved, the objective of the fact-finding task is to gather sufficient information about the business processes under consideration so that a design can be constructed which will then provide the blueprint for the systems build phase.

- **Project solving approach**

"The systems development life cycle is a problem-solving approach to building systems" (Whitten, Bentley & Barlow, 1994 p.92). The term problem is used here to include real problems, opportunities for improvement and directives from management. The classical problem-solving approach is as follows (ibid. p.92):

1. Identify the problem (or opportunity, or directive).
2. Understand the problem's environment and the problem's causes and effects.
3. Define the requirements of a suitable solution.
4. Identify alternative solutions.
5. Select the "best" solution.
6. Design and implement the solution.
7. Observe and evaluate the solution's impact. Refine the solution accordingly.

Systems analysts should approach all projects using some sort of problem-solving approach.

- **Designing systems for growth and change**

"Many systems analysts have fallen into the trap of developing systems to meet only today's user requirements. Although this may seem to be a necessary approach at first glance, it actually backfires in almost all cases" (ibid. p.96). Entropy is the term systems experts use to describe the natural and inevitable decay of all systems. Systems that are designed to meet only current requirements are usually difficult to modify in response to new requirements. The systems analyst is frequently forced to duplicate files and "patch" programs in ways that make the system very costly to support over the long run. As a result, many systems analysts become frustrated with how much time must be dedicated to supporting existing, patch-worked systems and how little time is left to work on important, new systems development (ibid.).

2.4 Design

"Design is the evaluation of alternative problem solutions, and the detailed specification of the final solution" (ibid. p.11). Throughout design, the emphasis usually shifts from the business to the computer solution. Design specifications are typically sent to programmers for systems implementation.

The systems design is directly constrained by the user requirements specification, which has been produced as a result of systems analysis. This will describe the functions that are required by the user which must be implemented as part of the design. There are also environmental constraints on design which are a result of the hardware and software environment of implementation. These include, hardware platforms, operating systems, data links required between the applications and other programs or a particular relational database, design tools such as case tools, methodologies or standards adopted by the organisations, system development tools such as programming languages, number of users to be supported and the performance required (Bocij et al, 1999 p.404).

2.4.1 User participation

In the design and development process of a new IS, the effect of user participation on system outcome is positive but the effect must be scrutinized by considering the contextual environment. It is necessary to include the relevant contingency factors like the impact of the system, the complexity of the system and the outsourcing of the system. These factors may affect user participation and system success both directly and indirectly (Lin & Shao 2000 p.292).

Good design happens only when designers understand people as well as technology. Designs that do not meet user's needs will often fail in the workplace or in the market, resulting in high costs in productivity, frustration and errors that impact users and their organisations (Smart & Whiting, 2001 p.178).

2.4.2 Multiplicity of user interfaces

Users of an Internet application could be using any browser such as Internet Explorer or Netscape. They might be using an old version of a browser or an old operating system like Windows 95. They might be using a brand new computer, but with a monitor set at an unexpected resolution. To make sure that a web site has consistent look and feel with equal performance, the user interface (UI) and application should be flexible to handle a variety of environments. The interface, for example, has to deal with various monitor resolutions to make sure that alignment of screens or appearances of graphics are not lost. Not all browsers uniformly support scripting languages like JavaScript, JSP, and ASP (Raveendra, 2001 p.2).

Therefore an application has to operate well with a variety of software and hardware considerations, not just the latest version of a browser. In addition, not all browsers conform to standards which adds more difficulties.

"One of the mistakes that developers make is assuming that the user has a good knowledge of working on a computer" (Smart & Whiting, 2001 p.178). Consider, for example, a users knowledge of working on an operating system like Windows. Sometimes the application is easier to work on if users know the shortcut keys (for example, ALT TAB) or users know what to do if certain situations arise. It is imperative that the system is easy to navigate without the users having to know shortcuts related to the operating system. If this is the case, then users should be trained properly. However, with a web application, most of the time users are not known. Therefore, it is imperative that the help facility, which explains the shortcut keys and what to do in certain situations, is readily available.

2.4.3 User interface design principles

The design of the user interface is key to ensuring that the software is easy to use and that users are productive. User interface design involves three main parts, first, defining the different views of the data such as input forms and output tables; second, defining how the user moves or navigates from one view to another; and third providing options for the user.

User interface design principles (Bocij et al, 1999 p.436)

1. *Functionality:* The main purpose of the user interface is to allow operators to complete their tasks effectively, quickly, easily and without frustration.
2. *Consistency:* Consistent systems are easier to learn and use because similar operations are performed in a similar manner in different modules. There needs to be consistency between applications which make it easy to use new applications.
3. *Navigation and control:* The way in which the system works and the way in which tasks and information are structured should be clearly revealed to the user. Users would be guided through the interaction process in the quickest and most efficient manner.
4. *Modes:* A mode is where the system only allows a restricted set of actions. Most force the user to focus on the way the system works rather than on the task at hand and should be avoided or clearly marked for the user.
5. *Relevancy:* It is important that only relevant and useful information is displayed.
6. *Visual clarity:* Users need to be able to find the information they require easily in order to interact with the system quickly. Each screen needs to be easy to read, uncluttered and the user's attention should be focused on important information. Important information needs to be highlighted to attract the user's attention.
7. *Feedback:* Informative feedback helps the user to understand what the system is doing and to determine exactly what is required next by the system.
8. *Terminology:* Every word and phrase that appears on the computer screen should be meaningful and helpful in the completion of the user's task. Technical terms and computer jargon should be avoided.

9. *Help:* Users should be encouraged to learn about the system. This will ensure that they are using full functionality. Users need to be able to use a help facility quickly and easily.
10. *Data input:* The user must be able to enter information easily and quickly. Fields should be formatted to cue the user to the type of information required. This will minimize potential errors. Validation of data input should occur.
11. *Error handling:* The system should be designed to minimize the possibility of user error. All user input should be validated before processing. The system should clearly and promptly inform the user when an error is detected and include information which will enable the reason for the error to be traced.

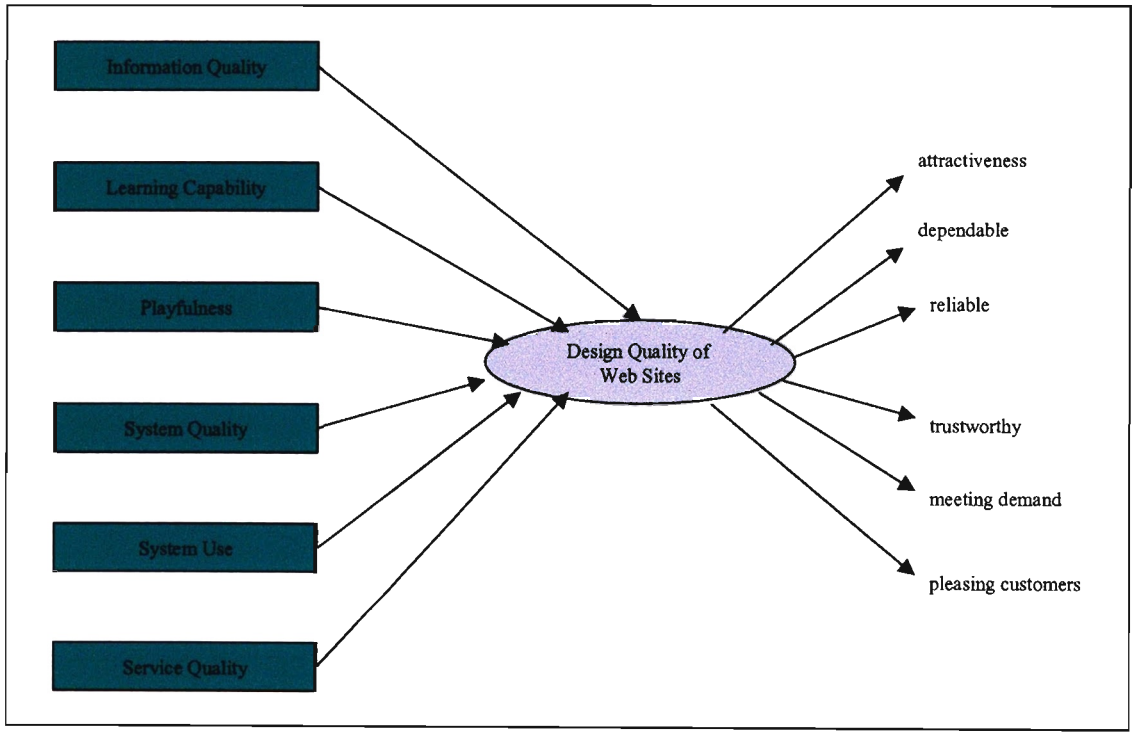
2.4.4 Criteria for developing a web site

Web sites are being widely deployed throughout industry, education, government and other institutions. Electronic commerce is a way of conducting business by companies and their customers performing electronic transactions through computer networks. Electronic commerce can help business organisations cut costs, interact directly with customers, run more smoothly and in a timely manner, and even better, it can help an organisation outperform its competition (Liu & Arnett, 2000 p.23).

A web site today is rarely a single machine that serves up pages. Increasingly, web sites represent distributed areas of network intelligence, web content and application services across networks to optimize web response time. Building a super-fast web site can be reduced to three architectural issues: caching, routing and load balancing. Web administrators must use a combination of these hardware and software techniques to optimize performance (Levitt & Harbaugh, 2000 p.1).

As the dependency on web technology increases, so does the need to assess factors associated with web site success. Customers would not pay for products or services over the web if financial information could not be transmitted securely - secure transactions are critical to the success. However, security is only a necessary but not a sufficient condition of designing a successful web site. The general definition of IS success is: the extent to which a system achieves the goals for which it was designed. A web site is a new type of information technology (Liu & Arnett, 2000 pp.23-25).

Fig 2: Design quality of web sites



Source: Liu & Arnett, 2000 p.26

Fig 2 shows that information quality, learning capability, playfulness, system quality, system use and service quality are factors that affect the design quality of a web site.

- *Information quality* involves accuracy, timeliness, relevance, flexible information presentation, customized information presentation, price information, product/service comparability, product/service differentiation, complete product/service description, perceived information quality on product/service, satisfying ethical standard and support business objectives.
- *Learning capability* involves well-organized hyperlinks, help function, customized search engine, interactive function between customers and businesses and interactive function among customers.
- *Playfulness* involves enjoyment, excitement, feeling of participation, escapism and charming.
- *System quality* involves rapid access (processing speed), quick error recovery, correct operation and computation, security, balanced payment method between security and ease of use and coordination to support all functional areas.
- *System use* involves, customer control of a transaction process, ease of use, confidence, tracking order status and privacy.
- *Service quality* involves quick responsiveness, assurance, reliability, empathy and follow-up service (ibid. pp.26-27).

Business organisations and web developers should (ibid. p.28):

- Actively seek ways to improve information and service quality provided through web sites.
- Establish a service-orientated concept to provide high quality service and high quality information.
- Focus on the way in which customers use a web site.
- Cultivate hedonic pleasures in the web site by motivating customers to participate, promoting customer excitement and concentration, and including charming features to attract customers and to help them enjoy the visit.

"Creativity must be incorporated into the design process in order to obtain customer psychological satisfaction when engaging in marketing on the web" (ibid. p.29).

Another issue which needs to be considered is the amount of traffic on the website which would affect the speed of the application. An organisation needs to do capacity planning so that the data centre infrastructure can be revamped to meet voluminous demands. Errors on a web site are another serious issue that needs to be addressed. The web application needs to be tested thoroughly before the users can work or test the system. When numerous errors occur during the testing phase, it reduces the users trust in the system (Raveendra, 2001 p.5).

Caching is the act of moving data – graphic files, HyperText Markup Language (HTML) code, or dynamic content – around the network to best meet the needs of network users. Databases, for instance, are always cached. But caching the results of database queries for web transactions is a complexity that no vendor has sufficiently solved. Static and dynamic web pages can also be cached. Typically, static web pages are served from one set of machines, while dynamic content, which usually includes database data, comes from another set of machines. It's possible to cache static web pages almost completely so that they get served ultra-fast (Levitt & Harbaugh, 2000 p.1).

The applications themselves are also a concern for speeding up the web site. If the application connects too often to a database, it can slow the application; therefore codes must also be optimized. Codes that are not properly debugged could mean that recovering from a crashed browser can take longer than waiting for a slow connection (ibid. p.1).

One of the key elements in input by all these methods is ensuring the quality of data. This is achieved through data validation. Data validation is a process to ensure the quality of data by checking that it has been entered correctly and prompts to the user informing them of incorrect data entry (Bocij et al, 1999 p.347).

2.4.5 Risks

In the design phase, the initial part is the Elaboration phase, and the latter is the Construction phase.

During Elaboration, the focus shifts to technology. The goal of the Elaboration phase is to achieve an architectural baseline by mitigating (mostly) technical risks. Mitigation is achieved by developing and testing critical parts of the system (i.e., through an architecture-first approach). Another risk is that the user will see an early version of the system for the first time, and typically only about 20% of it will work. To maintain buy-in from the user group, it's critical to manage expectations (Cardozo, 2002 p.8).

In the Construction phase, the focus in on completing the system, and the motto is "speed and quality". This can only be achieved by adding functionality to a stable architecture and having an effective build/release process. Both should be established in the Elaboration phase. When customers see the system in Elaboration, they might come up with new requirements. In addition, because more developers are brought in during Construction, communications can begin to break down within the project team. In this phase, the team must be focused on adding Use Cases to the systems. To overcome feature creep, they must negotiate any new requirements (by adhering to the business case) and ensure that an effective change process is in place (ibid. p.8).

2.5 Implementation and support

Implementation is the construction or assembly of the problem solution, culminating in a new environment based on the solution. Once implemented, the new system is said to be "in operation" or " in production" (Whitten, Bentley & Barlow, 1994 p.11).

2.5.1 Risks

In this phase, the focus is on acceptance testing and repairing defects. The project team must facilitate the acceptance process. Therefore, the release and change process must be quick and reliable. The deployment of new releases or fixes must be rapid to keep the acceptance-testing going. When repairing defects, the team must ensure that the integrity of the system is not jeopardized and that old defects do not return. The chances of introducing new defects when fixing known defects is high, and in this phase, the focus must be on repairing must-fix defects – those that impact primary Use Cases (i.e., that crash the system or make it unreliable or inaccurate). The project team should think twice before attempting to add missing functionality, and should not even think about adding "nice-to-have" features and functions. The project team should ensure that there is a change control board in place that includes a few project stakeholders. This board should classify all defects and authorize repair only of those they identify as "must-fix" (Cardozo, 2002 p.9).

2.5.2 Software change management

"A significant advantage of the phased approach to systems development is that it provides several opportunities to reevaluate feasibility" (Whitten, Bentley & Barlow, 1994 p.94). Many analysts allow project scope to increase during a project. Sometimes this is inevitable because the analyst learns more about the system as the project progresses. At each stage of a systems development project, change requests or variations to requirements will arise from business managers, users, designers and programmers. These requests include reports of bugs and features that are missing from the system as well as ideas for future versions of software.

Requests will occur as soon as users start evaluating prototypes of a system and will continue through to the maintenance phase of the project when the system is in production (Bocij et al, 1999 p.467).

The process of change needs to be carefully managed otherwise it can develop into *requirements creep*, a problem experienced in many information systems projects. As the numbers of requirements grow, more developer time will be required to fix the problems and the project can soon spiral out of control. What is needed is a mechanism to ensure first that all the changes are recorded and dealt with, and second that they are reviewed in such a way that the number of changes does not become unmanageable. Unfortunately, most analysts fail to adjust estimated costs and schedules as scope increases. As a result the project experiences cost and schedule overruns (ibid; Whitten, Bentley & Barlow, 1994 p.94).

The main steps in managing changed requirements are (ibid.):

1. Record the change requests, indicating level of importance and module affected.
2. Prioritise them with the internal or external customer as "must have", "nice to have" or "later release", in relation to the project constraints of system quality, cost and timescale.
3. Identify responsibility for fixing the problem, since it may lie with a software house, internal IS staff, systems integrator or hardware vendor.
4. Implement changes that are recorded as high priority.
5. Maintain a check of which high-priority errors have been fixed.

2.5.3 Resistance to change

Some resistance to change is inevitable, but this is particularly true with the introduction of systems associated with business process reengineering, since the way work is performed and people's job functions will be changed. If the rationale behind the change is not explained, then all the classic symptoms of resistance to change will be apparent.

There are many understandable reasons for people to resist the technological change that comes from the development of new information systems. These include social uncertainty, limited perspectives and lack of understanding, threats to power and influence of managers (loss of control), perception that costs of new system outweigh the benefits, fear of failure, inadequacy or redundancy (Bocij et al, 1999 p.476).

It is evident that training and education can be used to counter many of these issues. Additionally, other steps can be taken to reduce resistance to change, namely (ibid. p.476):

- Ensure early participation and involvement of users.
- Set realistic goals and raise realistic expectations of benefits.
- Build in user-friendliness to the new system.
- Don't promise too much and deliver what was promised.
- Develop a reliable system that is easy to maintain.
- Ensure support of the various stakeholders.
- Bring about agreement through negotiation.

2.6 Project failures

The reasons IT projects fail, to whatever extent, remain the same as always: "... the inability to specify user requirements, managing the number of requested changes or limiting the scope of change as the project progress" (ibid. p.308). There are many other issues behind failed projects, including in-house politics, deadline-centric cultures and new legislation emerging during the project. The most commonly reports causes of information systems project failures are as follows (Hartman & Ashrafi, 2000 p.6):

- Misunderstood requirements
- Optimistic schedules and budgets
- Inadequate risk assessment and management
- Inconsistent standards and lack of training in project management
- Management of resources
- Unclear charter for a project
- Lack of communication

In detail this includes (Lientz & Rea, 1999 p.12, Bocij et al, 1999 p.308):

1. Major stakeholders generally do not have a clear idea of project success or have differing views of what success constitutes. If a clear vision exists, it is not effectively communicated or the project team does not understand it. This leads to scope creep, inappropriate measurement, churn in developments, specification changes, delays and other issues. Project goals are not understood or agreed on. There is sometimes a lack of understanding and agreement about goals of the project. Business, technical, and organisational objectives might overlap and conflict.
2. The scope of the project is not well defined until substantial work has been done. Additional requirements that surface then enlarge the scope.
3. Generally there is a problem in identifying key result areas (KRAs) and critical success factors (CSFs) and linking them to the stakeholders' business strategy. This leads to lack of support by senior management.
4. The project team and major stakeholders are not very clear on what the performance and control metrics should be. Normally the focus is on time, cost, performance and quality. But this focus is not consistent between stakeholders or over time. Some have recognised the importance of customer and end-user satisfaction.
5. Project control and performance metrics are not linked to KRAs and CSFs. This means that one measures the wrong things and distracts the team from what is important to success. It looks like inadequate or ineffective project control.
6. Generally there is very little, or sometimes, no alignment among major stakeholders on success criteria, KRAs, CSFs, performance metrics, project drivers and on the dynamics of change for these elements over the project life cycle. This leads to inappropriate decision-making and inconsistency in management style and focus.
7. The project team is weak. The project team is weak or lacks technical knowledge and experience. No provision is in place for assigning senior people to the project.
8. *Technical failure* stemming from poor technical quality.
9. *Data failure* due to (a) poor data design, processing errors and poor data management; and (b) poor user procedures and poor data quality control at the input stage.

10. *User failure* to use the system to its maximum capability – may be due to an unwillingness to train staff or user management failure to allow their staff full involvement in the systems development process.
11. *Organisational failure*, where an individual system may work in its own right but fails to meet organizational needs as a whole (for example, while a system might offer satisfactory operational information, it fails to provide usable management information). This results from senior management failure to align IS to overall organization needs.
12. *Failure in the business environment.* This can stem from systems that are inappropriate to the market environment; failure in IS not being adaptable to a changing business environment (often rapid change occurs), or a system not coping with the volume and speed of the underlying business transactions.

It is apparent that a diverse range of problems can cause a project to fail, ranging from technical problems to people management problems. "It is the responsibility of the project manager to ensure that these types of problems do not occur, by anticipating them and then taking the necessary actions to resolve them. This will involve risk management techniques" (Bocij et al, 1999 p.308).

2.7 Critical success factors

Hartman and Ashrafi (2000 p.6) believe that the reasons identified for project failure are symptoms of the disease and not the root causes of the disease. They believe that the CSFs are the elements that make a project a success. These include trust, effective communication, top management support, etc. KRAs are specific results that are needed to deliver a successful project. CSF methodology has been highly successful in identifying KRAs crucial for the success of a project.

With changing business conditions, half-century-old project performance metrics are no longer effective for the monitoring and control of today's projects. Proper measurement tools and metrics are necessary for effective control of projects.

Table 1 shows the Critical Success Factors that will make a project successful.

Table 1: 10 most important critical success factors and metrics

Rank order	Critical success factors	Project metrics
1	Owner is informed of the project status and his/her approval is obtained at each stage	Project completed on time or ahead of schedule
2	Owner is consulted at all stages of development and implementation	Milestones are identified and met
3	Proper communication channels are established at appropriate levels in the project team	Deliverables are identified
4	The project has a clearly defined mission	The scope of the project is clearly defined and quantified
5	Top management is willing to provide the necessary resources (money, expertise, equipment)	Activities and logical sequences are determined and scheduled
6	The project achieves its stated business purpose	Project completion is precisely defined
7	A detailed project plan (including time schedules, and milestones) with a detailed budget in place	The project is completed within a predetermined budget
8	The appropriate technology and expertise are available	Resource requirements are identified and supplied as needed
9	Project changes are managed through a formal process	Responsibilities are assigned
10	The project is completed with minimal and mutually agreed scope changes	A specific new technology is adopted and accepted by end users

Source: Hartman & Ashrafi, 2000 p.12

Other factors contributing to the success of a project that needs to be considered (Lientz & Rea, 1999 p.16):

- Team building: By having team members more active in project management they have a greater sense of commitment. The same is true for business units. Project leaders are encouraged to work together in a cooperative mode.
- Issue orientated: This process makes management more issue orientated rather than status orientated. Issues that cannot be resolved by individual or multiple project managers are considered by management.
- Overall focus on projects: Because the projects are based on major business processes that cross multiple departments, the projects will consume more of the available resources. This means that many of the smaller, enhancement type projects will fall by the wayside due to a lower priority.
- Impact on information systems: with this new project focus, information systems become a key supporter, if not the owner, of processes that involve multiple departments.

Recommendations (Hartman & Ashrafi, 2000 p.12):

- Link your project to corporate business strategy
- Align major stakeholders on key issues
- Simplify project controls and metrics
- Make sure effective communication and expectation management is maintained throughout the project life.

However, if the owner, contractor, and consultant on a project all have different ideas of what success is and how success will be measured, it is unlikely that everyone will be satisfied when the project is completed.

2.8 Conclusion

This chapter focused on understanding the best practice for developing an information system. The SDLC was the framework used which includes fives phases, namely, Planning, Analysis, Design, Implementation and Support. Best practices were determined for each phase of the SDLC. Reasons for project failure and critical success factors were also summarized in this chapter to highlight problem areas in all the phases. Chapter three explains how the research was conducted. Chapters four and five evaluates the system developed by the financial institution and was based on the best practices identified in this chapter.

CHAPTER THREE: PLANNING OF THE RESEARCH

3.1 Introduction

Schumacher and McMillan (1993 p.8) define *research* as a systematic process of collecting and logically analysing information (data) for some purpose. This definition is general because there are many methods available to investigate a problem or question. *Research methods* (sometimes called "methodology") are the ways one collects and analyses data. These methods were developed for acquiring knowledge by reliable and valid procedures. Data collection must be done with measurement techniques, extensive interviews and observations, or a collection of documents.

Research methodology is systematic and purposeful (Van den Aardweg & Van den Aardweg, 1988 p.197). Procedures are not haphazard activities; they are planned to yield data on a particular research problem. In a broader context, methodology refers to design whereby the researcher selects the data collection and analysis procedures to investigate a specific research problem. It is possible to have a design that provides no valid or reliable data on the problem, but the deliberate choice of a design increases the likelihood that the data will yield information on the research question.

Cronbach and Suppes (1969 pp.15-16) further suggest that whatever the character of the study, if it is disciplined, the investigator has anticipated the traditional questions that are pertinent. He/she institutes control at each step of information collection and reasoning to avoid the sources of error to which these questions refer. If the errors cannot be eliminated, it is taken into account by discussing the margin of error in the conclusions. Thus, the report of a disciplined inquiry has a texture that displays the raw materials entering the argument and the logical processes by which they were compressed and rearranged to make the conclusion credible.

Wiersma (1991 p.8) identifies five steps that characterise the systematic nature of the research process. These are: (1) identifying the problem, (2) reviewing information, (3) collecting data, (4) analysing data, and (5) drawing conclusions. This chapter is about the third step, i.e. collecting data.

Aspects under discussion in this chapter include preparation and design of the research, permission, selection of respondents, the research instrument, the pilot study, administration of the questionnaire, processing of the data and limitations of the investigation.

3.2 Preparation and design of the research

According to Schumacher and McMillan (1993 p.31) *research design* refers to the plan and structure of the investigation used to obtain evidence for conducting the study, including when, from whom, and under what conditions the data will be obtained. In other words, design indicates how the research is set up: what happens to the subjects and what methods of data collection are used.

The project under investigation involves a survey research. Schumacher and McMillan (1993 p.36) explains that in a research survey, the investigator selects a sample of subjects and administers a questionnaire or conducts interviews to collect data. Surveys are used frequently in educational research to describe attitudes, beliefs, opinions, and other types of information.

Usually the research is designed so that information about a large number of people (population) can be inferred from the responses obtained from a smaller group of subjects (sample).

3.2.1 Permission

The project team and head office users of the system were personally interviewed and the branch users of the financial institution were telephonically interviewed. Written permission was received by the financial institution's IT Department as it served as the source of this research study.

3.2.2 Selection of respondents

a) Sampling

Slavin (1984 p.98) observes that one very important aspect of research design, especially in survey research, is the determination of the appropriate sample. As the word implies, a sample is a part of a larger whole

Ary, Jacob & Razavieh (1979 p.138) point out that inductive reasoning is the rationale of sampling. The inductive method involves making observations and then drawing conclusions from these observations. This is the concept of sampling, which involves taking a portion of the population, making observations on this smaller group, and then generalising the findings to the large population. It is extremely important that the individuals included in a sample constitute a representative cross section of individuals in the population. That is, samples must be representative if one is to be able to generalise with confidence from the sample to the population.

Sowell and Casey (1982 p.75) states that there are four basic types of scientific sampling methods, namely: simple random, stratified random, cluster and systematic sampling.

b) Purposive sampling

According to Cooper and Schindler (2001 p.192) purposive sampling is a nonprobability sample that conforms to certain criteria. There are two major types: judgement sampling and quota sampling. Judgement sampling occurs when a researcher selects sample members to conform to some criterion. Quota sampling is used to improve representativeness. The logic behind quota sampling is that certain relevant characteristics describe the dimensions of the population. For the purpose of this study, judgement sampling was used. Members with different roles in the project team were interviewed. Branch representatives and project owners were also interviewed.

c) The size of the sample

Wiersma (1991 p.264) observes that a number of factors may affect the sample size. In educational research, available resources of time, money, personnel and facilities are often the most influential. Generally, increasing sample size enhances statistical precision. However, it should not be inferred that it is always desirable to increase the sample size to its maximum, since this may be unduly costly and wasteful of effort and information. According to Gay (1987 p.114) for descriptive research, a sample of 10% of the population is considered minimum. For smaller populations, 20% may be required.

For the purpose of this research, the following people were interviewed:

- The project manager
- Two project leaders
- Two business analysts
- The database administrator
- Three developers
- One project owner
- One branch representative from each branch

3.3 The research instrument

Cooper and Schindler (2001 p.292) claim that there are basically six ways to collect data: observations, questionnaires, interviews, documents, tests and unobtrusive measures. All research uses a variation of one or more of these, depending on strengths and limitations of each and other considerations. The research instrument that best served the needs of this research study was interviews.

3.3.1 The interview

A personal (i.e., face to face communication) and telephonic interview is a two-way conversation initiated by an interviewer to obtain information from a respondent (ibid. p297).

a) Personal interviews

i. **Advantages** (ibid p.313)

- Good cooperation from respondents.
- Interviewer can answer questions about the research, probe for answers, use follow-up questions, and gather information by observation.
- Special visual aids and scoring devices can be used.
- Interviewer can prescreen respondent to ensure he/she fits the population profile.

ii. **Disadvantages** (ibid.)

- High costs.
- Longer period needed in the collecting data.
- May be a wide geographical dispersion.
- Not all respondents are available or accessible.
- Some respondents are unwilling to participate.
- Questions may be altered or respondent coached by interviewers.

b) Telephone interviews

i. Advantages (ibid.)

- Lower costs than personal interviews
- Expanded geographic coverage without dramatic increase in costs
- Reduced interviewer bias.
- Fastest completion time.
- Better access to hard to reach respondents through repeated callbacks.

ii. Disadvantages (ibid.)

- Response rate is lower than personal interview.
- Higher costs if the interviewing geographically dispersed sample.
- Many phone numbers are unlisted or not working, making directory listings unreliable.
- Some target groups are not available by phone.
- Responses may be less complete.
- Illustrations cannot be used.

3.3.2 Construction of interviews

Personal interviews were conducted with the project team and project owners as they were easily accessible due to them working all at head office. Telephonic interviews were conducted with the branch representatives due to the geographical location. Due to the personal relationship between researcher and respondents, a questionnaire did not seem feasible. The researcher also did not have to be concerned about non-responses from questionnaires. There were open-ended questions asked that involved the interviewer to ask follow-up questions based on answers received.

Appendix A, B, C, D and E shows the different questions asked to the various people. The interviews were more conversational in nature. Questions in the appendixes were used as a guideline to ensure that the interview was constructive and all information needed for the research was received.

Cooper and Schindler (2001 p.333) identified three types of measurement questions, namely, administrative, classification and target questions. Administrative questions identify the respondent, interviewer, interview location and conditions. Classification questions are usually sociological-demographic variables that allow respondent's answers to be grouped so patterns are revealed and can be studied Target questions address the investigative questions of a specific study.

For the purpose of this research, it was not necessary to ask administrative and classification questions as this information did not impact of the research. Target questions were asked, however, the interview was conversational. The interviewer encouraged the respondents to talk in-depth about certain aspects of the project. The in-depth interview encouraged respondents to share as much information as possible in an unconstrained environment.

3.4 Pilot study

The pilot study, sometimes referred to as pilot testing, is a preliminary or "trial run" investigation that precedes the carrying out of any investigation or project (Cooper & Schindler, 2001 p.81). The basic purpose of a pilot study is to determine how the design of the subsequent study can be improved and to identify flaws in the instruments, for example, questionnaires or textual materials, to be used. The number of the participants in the pilot study or group is normally smaller than the number scheduled to take part in the subsequent study.

In the pilot study, the researcher tried out a number of alternative measures and then selected those that produced the best results for the main study. The interview questions were submitted to two qualified academics to ensure that questions are free from bias. Therefore, pre-testing and pilot study provided guidance in the present study on the suitability of questions and valuable supporting evidence.

3.5 Administration of the interview

Cooper and Schindler (2001 p.302) suggests that researchers may find it useful to mail an introductory letter to the respondents before scheduling an appointment. This alerts the subject to the study rather than overwhelm them at the interview.

The researcher did not, however, follow Cooper and Schindler's (ibid.) suggestion to the letter, but used their suggestion as a guide in the administration process. Due to the personal relationship between the researcher and respondents, a brief introduction was given to each respondent explaining the aims and objectives of the research study at the start of the interview.

3.6 Conclusion

Chapter three serves to outline the criteria and procedures that the researcher had to consider in the planning of the research. Planning incorporated permission, selection of respondents, the research instrument (in this case, interviews), the pilot study and how the interview was administered. All the above aspects served in the construction of a credible research design.

From the interviews conducted, it was found that there were no discrepancies in answers received from all respondents. Chapters four, five and six are based on information received from the interviews.

CHAPTER FOUR: REVIEW OF THE PROJECT

The financial institution took a decision to provide additional cash services to the branches and their clients. The IT Department established the project to provide the relevant cash management system. The reason for the initiation of the project was due to the reports from the institution's Internal Audit Department about the current cash management system. Their findings showed that the current system was unstable at times and inadequate for future enhancements like, for example, integrations with other systems and new technologies.

4.1 Planning

Focusing on costs and control, the following options were considered by the project management team, which included the project manager and project leader:

- Enhancing the existing cash management and other systems in the financial institution.
- Buy an "off-the-shelf" system.
- Implement a system developed by national/international solution providers and the IT Department.
- Implement a system developed by national/international solution providers only.
- Implement a system developed by the IT Department only.

The option of enhancing the two existing cash management systems was not seen as a long-term solution, although these systems could assist in bridging the short to medium term requirements. These systems were old, inadequate, could not integrate with other systems and even unstable.

Information gathered indicated that "off-the-shelf" solutions for the cash management system requirements do not exist. Systems are being built, either in-house on the normal IT development platform or by external solution providers using specific tools and development platforms (tools developed either by themselves or by a third party). Support and maintenance of externally built systems are usually expensive and provided by the external solution providers.

A system developed by national/international solution providers without the IT Department was also not seen as feasible, mainly due to the amount of time that the IT Department will have to spend transferring the requirement information, the loss of control over the system and in the international solution provider case, the worsening exchange rate scenario.

The "IT Department only" solution was then proposed as being the most flexible solution and the most cost effective. It also provides the cost control in terms of security, future support and maintenance of the system.

Therefore, the primary objective of the project is to enable the management of bulk cash on a national level of redesigning, rewriting, enhancing and integrating the current cash management systems and providing additional functionality to interact with external parties.

4.1.1 Project management

The project management team held regular meetings to discuss the various issues facing the project. Risks were identified, assessed and documented. The project management team set the milestones, which were the major deliverables for the project, and was submitted to IT management and Internal Audit. The financial institution's head office and the seven branches throughout South Africa will be using the system. A department in head office is in charge of the operations of the seven branches and was therefore regarded as the owner of project. Cost and feasibility analyses were done and presented to the management team of this department.

a) Establishment of standards for consistent development and documentation

i. Use of unified tools

Lengthy discussions were held by the project management team to discuss what tools would be used for the development of the project. It was established that the system was to be a web based system due to the number of users and the geographical location of the users. Also with the focus of including external parties, who are the clients of the branches, to use the system in future, a web application seemed feasible for easy access to the system and system expansion. It was decided that the system be a three-tier application, where there is a front-end accessed by the users, a database where all information is stored, and a middle tier where stored procedures be written that will read and update the database and will consist of all the business rules.

All user requirements were to be documented using Use Cases. "A Use Case defines a goal-oriented set of interactions between external actors and the system under consideration" (Bredemeyer, 2000 p.1). *Actors* are parties outside the system that interact with the system. An actor may be a class of users, roles users can play or other systems. Bredemeyer (ibid.) distinguishes between primary and secondary actors. A *primary* actor is one having a goal requiring the assistance of the system. A *secondary* actor is one from which the system needs assistance.

A Use Case is initiated by a user with a particular goal in mind, and terminates successfully when that goal is satisfied. It describes the sequence of interactions between actors and the system necessary to deliver the service that satisfies the goal. It also includes possible variants of this sequence, for example, alternative sequences that may also satisfy the goal, as well as sequences that may lead to failure to complete the service because of exceptional behaviour, error handling, etc. The system is treated as a "black box", and the interactions with the system, including system responses, are as perceived from outside the system.

Thus, Use Cases capture *who* (actor) does *what* (interaction) with the system, for what *purpose* (goal), without dealing with system internals. A complete set of Use Cases specifies all the different ways to use the system, and therefore defines all behaviour required of the system, bounding the scope of the system.

Generally, Use Case steps are written in an easy-to-understand structured narrative using the vocabulary of the domain. This is engaging for users who can easily follow and validate the Use Cases, and the accessibility encourages users to be actively involved in defining the requirements. Use Cases have not been used in any IS development in the financial institution, but due to the geographical location of users, it seemed feasible to use.

The development team was to use Extensible Markup Language (XML) as this was the standard being used by most web applications so that the system can integrate with other systems should it need to in the future. "XML provides a flexible way of expressing and presenting data and allows data to be updated without having to refresh the entire page" (Lawrence, Newton, Corbitt, Braithwaite & Parker 2002 p.3). Microsoft Sequel Server was the database chosen for the system as there is adequate security built in the database and stored procedures are built into the database structure.

Crystal Reports 7 was to be used to design the reports for the system. This software package was used by several other systems in the financial institution and those users were comfortable with using the package.

ii. Project methodology

Due to the size of the project, the project was split into versions where the first version would just replace the current cash management system. The project management team decided on following an iterative approach to develop this version. This means that the system was broken down into several phases and within each phase was several iterations which represented the various functions. The business analyst had to get an overall understanding of the whole system in order to appropriately split the system into phases. Each iteration was designed, developed and tested by the IT Department before allowing the user to test. While a phase was being developed, the analyst already started receiving detailed user requirements for the following phase.

The project team set up three servers. The development team used the Development server to develop the system, the testing by the IT Department and users were performed on the Quality Assurance (QA) Server and the Production server.

iii. Establishing standards

Documents were stored on a server instead of a person's computer so that all the project team members would be able to access documents. All the Use Cases were stored on this server. Any changes to the user requirements were immediately updated on the Use Case. The Use Cases had to be updated on a regular basis to ensure that it stayed up-to-date. The project management team also kept documentation of meetings and investigations done.

The development and programming standards were discussed with the development team to ensure that the developers followed the same standard and style of programming.

b) Attacking risks

The project management team identified several risks, and due to this, the management of the project was to be risk-focused. Risks were identified and discussed at length to determine the action plan for each risk.

1. The unavailability of expert user knowledge – i.e. no backup for current user expert.

Action plan: Backup resource was to be confirmed and the backup was to be informed about his/her involvement/responsibilities.

2. Negotiation and obtaining agreement with external parties.

Action plan: Investigate existing communication structures and forums and determine the feasibility of using them. Establish a single point of contact between external parties.

3. Scope creep due to undefined or different expectations and branch operational procedures.

Action plan: On receipt of new business requirements submitted by the users, the impact should be determined and discussed with the project owners. They should make the final decision of whether the new business requirement is critical to the system or whether the requirement can be developed at a later stage.

4. Development environment be offline due to a virus attack/unforeseen circumstances.

Action plan: Regular backups of work should be enforced, and have an alternative development environment ready should the active environment go offline.

5. Data integration between systems.

Action plan: Various interfaces should be identified, and developed using open standards, i.e. use XML to be able to integrate with other systems.

The project plan, business specifications and systems specification documents were created to get user buy-in and support. The risk of receiving incomplete requirements also exists due to the geographical location of the users. Due to this, the analyst team visited the branches to understand how each branch operates.

c) Developing the 'delivery habit'

Each iteration was assigned a deadline which was documented and communicated to the development team. The database administrator (DBA) completed the preliminary design of the whole system. A group of DBAs then completed the stored procedures and the developers developed the front-end and used the stored procedures to read and update the database. Each developer in the team was given various functions to complete. The business analyst was creating Use Cases and once finalized by the users, would then be given to the developers. This ensured that the developers were not sitting idle waiting to develop functions. If one function could not be completed due to unforeseen circumstances, the developer can work on another function.

d) Scope management

Due to traditional software development approaches focused on functions instead of business processes, the project management team opted to use Use Cases to tie the system together. With the Use Cases, project stakeholders would be able to understand the requirements because it was documented in a language that the customer could understand. By doing this, users were able to inform the analyst during the initial stages of the project if the requirements are incorrect or incomplete.

To ensure that all project stakeholders were aware of the project scope, the project plan was completed which included the goals of the project and the goals of each version. Any new requests had to be discussed first with the project owners to ensure that it was within the scope of the system prior to completing the request.

e) Project management methodology

i. Project plans

The project plan was written which consisted of the main activities within the project. The plan also provided for an overall schedule and identified the resources needed and budget analysis. The goals of each version of the project were documented along with a breakdown of each phase within the current version and each iteration within a phase.

There were also individual work plans where iterations were assigned to developers. With this plan team members knew each other's responsibilities and deadlines.

The first and second phase consisted of all administrative functions. The third and fourth phases consisted of the core functionality of the system. This implied that all functions in the third and fourth phases could make or break the system.

ii. Project and management control

To ensure business integrity, schedules were set for each phase and iteration. In this way, the project management could keep track of the status of the project by comparing performance with the schedules. To ensure technical integrity, each Use Case had a Test Case which was to be used as a guideline for testing the functionality of the Use Case and system quality. CSFs were also determined to ensure that the project team has some criteria against which to measure the system progress.

Regular meetings were held to ensure that the system was progressing according to schedule. If a particular function was behind schedule, reasons for this was discussed and new were deadline set. However, the major milestones, like system implementation of the project were not rescheduled. These meetings were also held to handle any conflicts that arose within the development team. This was an opportunity for the development team to raise any of their concerns regarding the project.

4.1.2 User Involvement

All project stakeholders were identified. Head Office users would be using the system more for managerial reports and the branches would use the core functionality of the system. The project management team asked the project owners to nominate a person from their department to be a project owner representative to work full time with providing user requirements to the business analyst. They were also asked to nominate a branch representative to liaise with all the other branches to confirm their requirements for the system that will then be communicated to the business analyst. The two nominees were then informed of their responsibilities.

Requirements were documented using Use Cases by the business analyst and sent to the project owner and branch representative. The branch representative distributed the Use Cases to a representative in the other branches and awaited feedback. The branch representative would then consolidate the feedback received from the branches and add their comments to the Use Cases. These will then be sent back to the analyst to update the Use Cases.

In this way, the project team was trying to get the user involved from the initial phases of the project. Also, the branch representative dealt with the discrepancies from the feedback received by the other branches. Should the branch representative not be successful with any discrepancies, this would then be reported to the project owner representative to resolve. In this way business issues were left solely to the users to resolve.

Both representatives signed off the Use Case once satisfied that their requirements were properly documented.

4.2 Analysis

Each branch manager and a person from each branch who had an in depth knowledge of the current cash management system were brought to the head office to discuss their requirements with the business analysts. The analyst team then visited each branch to understand the operations of all the branches as each branch could work differently. The system was going to be "one system for all" and the analyst team needed to ensure that the system met the requirements for all users. The analyst team also had discussions with those who were involved with the development of the current cash management systems. They then documented the user requirements using Use Cases.

Each business process was designed as one Use Case. The Use Cases consisted of a brief description of the process and detailed flow of events. They also consisted of technical description of the database and all changes, both business and technical, were added to the Use Case. Some Use Cases had screen layouts drawn. The Use Cases were then e-mailed to the project owner and branch representatives. They provided feedback, and in some cases, workflow diagrams had to be drawn as there were several misunderstanding between users and analysts. Once these were finalized between the representatives and analysts, the branch representative e-mailed the other branch representatives and awaited feedback within a specified period.

Unfortunately, not many branches provided feedback. The Use Cases were then signed off and development started to take place. There were several changes made to the Use Cases after the users signed them off. This caused database design and program changes to take place. User comments were then added to the Use Cases, and certain information that was not applicable to the Use Case was struck off the document. The Use Cases therefore became very lengthy. After a while, updating of Use Cases became too tedious, as there were numerous changes that were being requested by the users and therefore updating was ignored.

Test Cases were designed for each Use Case and for the first two phases of the project, these test cases were used to test the system thoroughly before the users tested it. Due to time constraints, there were no test cases designed for the third and fourth phases and therefore, the system was not tested prior to users testing it.

4.3 Design

There was a preliminary database design done prior to finalisation of user requirements as the database administrator had knowledge of the system due to working in a branch. However, several changes had to be made to the database design after user requirements were being finalized. After the representatives signed off the Use Case, changes were still being made to the Use Case or additional functionality was being requested. Therefore the DBA's had to make additional changes to the database design and stored procedures. However, with the exception of one DBA, the other DBA's did not have any experience in writing stored procedures. Eventually, all the stored procedures were written or rewritten by the one DBA.

The project team who understood the whole system designed the system architecture. The menu system was dynamically created depending on what functions a user has been allocated. The system architecture took an extremely long time to finalise and this impacted on the future deadlines of the system as the development of the system was dependent on the architecture.

Users were not involved in the design phase. The developers worked solely from the Use Cases developed in the analysis phase that changed during the design phase. Due to these changes, developers had to change their code to suit changing requirements. Once the development team completed the Use Case, the tester would test the application in the QA environment using the Test Cases designed for each Use Case. If the system did not do what the Use Case stated, the Test Case would be marked as failed.

The first two phases, which were mainly administrative functions, did not make their deadlines due to the fact that the architecture of the project took long to finalise and user requirements were changing. This had caused a delay in the two phases to follow. Further, due to the tight schedule, the last two phases were then combined into one phase. Due to the lack of resources, developers were assigned several functions to be completed in a short space of time. And finally, due to time constraints, the system could not be thoroughly tested like it was in phases one and two.

4.3.1 Multiplicity of user interfaces

Due to the system being an internal system, Internet browser standards were set such that users had to have a minimum of IE 5.5 with Service Pack 2 installed. The system used Secure Sockets Layer (SSL) to transmit transactions to the database. Therefore all user computers needed to have 128-bit encryption. Another standard was that computer resolutions should be at least 800 by 600 pixels. For security reasons, the application was to open in full screen, without the option of minimizing/closing without logging off.

4.3.2 User interface and web site design principles

The project team discussed the user interface design in great length. The system architecture, which included how the menu will be displayed and how certain processes were to be handled by the system, was discussed. The menu structure was designed so that the users would be able to easily navigate through the menu. The system was designed to ensure that users were able to complete their tasks effectively and efficiently. It was decided to have a TASK LIST so users would be able to easily identify the status of functions, i.e. the task list will display all transaction that still needed authorizations. The task list avoids users having to click on different functions to verify or authorize transactions. With the new system there was just one central point that users had to go to.

Standards were also set to ensure that screen layouts and user navigations were consistent throughout the system. The terminology used in the system is the same terminology used by the current system so users would not get confused. The help facility is readily available and explains each function in detail from the purpose of the function to how the function should be completed.

The system also made use of drop down lists which makes it easier for the user to see the options available and selecting the appropriate one. Users are able to capture information using input boxes. There is also information on the screen to inform the user as to what format the data should be in. The TAB button also allows easy scrolling from one input box to another. Date fields have a calendar that the user must select to avoid wrong date formats. There are also user and business rule validations included in the program before being updated to the database to avoid incorrect information being updated to the database causing integrity errors and data to be unreliable.

A network connectivity analysis was done to investigate the response time from the branches. The investigation showed that the network connectivity is at an acceptable level. The programs use very few connections to the database to avoid slowing the response of the system. Transactions are transmitted securely over the web using SSL to ensure that the transaction cannot be tampered or changed by someone intercepting the transmission before the transaction reaches the database at Head Office.

When a phase was completed and tested by the project team, the users were asked to test the system in QA. Users were informed about what they needed to test and the procedure that they should follow. The users who started working on the system for the first time were requesting several changes even in phase one. Developers had already started working on the next phase, but had to go back to the previous phase functions to make the necessary changes.

4.4 Implementation and support

Once the last phase of the system was completed, two representatives from each branch visited Head Office to be trained on the new system. The users were given an opportunity to test the system in phase one and two, but due to phase three and four being combined, the core of the system was designed all at once. Therefore the users had to be trained as there were several functions developed and they needed to be shown how to use the new system. Due to this date being set at the start of the project, there was no time for the project team to test the system. The first day of the training did not go very well as the system was not doing what the users wanted. Also, there were errors in the system that did not make a good impression on the users. The next day, it was decided to rather explain the processes and demonstrate the system. In this way, users were able to interpret what was discussed and make comments. Several changes were requested to the screen design and processes.

Thereafter the users had an opportunity to test the system for two weeks before the system went parallel with the old system. However the system still had several errors and functional faults due to the time constraints and incomplete user requirements that the parallel run had to be rescheduled for a month later. During this time, there was an influx of calls and errors logged and changes being requested, as the users could not work with the way the system was developed.

Developers were continuously making changes to programs as faults were being reported and new requirements were asked for. Due to many functions not working the way the users wanted it to work, the parallel run had to be moved by another month to meet user requirements. The main problems were then the reports that did not meet user requirements.

During the parallel run the project owners started to work on the system to do authorisations. Some had their own requirements in mind, which were not met by the system. They then requested additional business rules, some of which were completed and others could not due to the requirements involving major changes and not within the scope of the project.

4.5 Conclusion

From the review, it is evident that the system started running over schedules at the start of the project which continued throughout the development. It is also apparent that users requirements continuously changed even after signing off the Use Cases that contributed to the project missing its deadlines and milestones. Chapter Five will explain why the project did not meet its deadlines and milestones by evaluating what was done against the best practices identified in the Chapter Two.

CHAPTER FIVE: EVALUATION OF THE SYSTEM'S DEVELOPMENT

5.1 Planning

The project team investigation of cost and feasibility of the project was performed well. It is always necessary to ensure that one is not "reinventing the wheel", i.e. developing a system that has already been developed. The project team did a thorough investigation to ensure that there is no cash management system that already exists to meet the institution's requirements. However, these investigations were not properly communicated to the project owners. There was a project owner who believed that there was an "off-the-shelf" package that could have been used instead of wasting time developing the system from scratch. These packages were investigated by the project management team and were found not to meet the all system requirements. It was apparent that the project owner did not understand all the requirements of the system.

5.1.1 Project management

The project milestones that were set were reasonable at the time of submission. During the development, however, these milestones seemed unrealistic as phases and iterations of the project missed the deadlines set by the project team. The project milestones were not pushed forward when the project started experiencing deadline problems.

a) Establishment of standards for consistent development and documentation

i. Use of unified tools

The decision to use the web for the system was very good. With the future plans of the project, the web infrastructure was suitable to meet the requirements. The Internet is changing business much faster and in more far-reaching ways than one could have predicted. "The potential of cutting costs by conducting business via the web is enormous" (Daum & Horak, 2001 p.5)

The decision to use XML was a key to the success of the project as XML is "a universal meta language that ensures that Internet applications 'understand each other' and that Internet applications and traditional enterprise software can communicate smoothly" (ibid. p.5). XML makes it much easier to design integrated business processes, i.e. storing, publishing and exchanging of electronic documents is made much easier using XML (ibid. p.6).

It was decided to use Crystal Reports 7 for the reports in the system. Even though this package was being used by other systems in the financial institution, due to this system being very large, it was only discovered during the parallel run that the current version of Crystal Reports 7 did not perform well. Some reports caused some user computers and printer servers to "hang" and other reports were displaying errors related to the software. Therefore Crystal Reports 9 had to be purchased, but this did not cost the institution anything, as there was a service agreement made with the Crystal Reports vendor for updates. However, all reports had to be upgraded to the new version and each program that used a report had to be changed and this resulted in a lot of time being wasted.

The use of Use Cases was decided for the analysis of the project. This was the first time that Use Cases were used for any project in the financial institution. However, due to the advantages discussed for using Use Cases and the geographical location of users, this decision was feasible. The business analysts and developers used the Use Cases and updated them to ensure validity of Use Cases. This was done well up to a point where there was no pressure evident which is normal during the early stages of development. When users started reporting a lot of errors and functional faults, there was very little time to fix the errors and faults and retest before informing users to retest. Due to this, Use Cases and Test Cases were not updated.

The project team did not investigate the users' knowledge of working on a web application. This should have been done, as the current cash management system was a mainframe application. Users were expected to give feedback of user requirements based on the information in the Use Case. However, users were not trained to use the Use Cases.

ii. Project methodology

The system was split into four phases with the first two being administrative functions and the last two being the core functionality of the system. Due to the development team spending too much time on the administrative functions, the core functionality was completed in less time than it did to complete the administrative functions. Therefore the project team did not have time to test the core functions of the system. This should not have been done, as any errors that occur could have been identified before the user started using the system. There is a better chance of users accepting and trusting the system if the system does not have any errors.

The idea behind splitting the system into phases was to expose the users to the system in small logical steps. However, due to time constraints the system was eventually delivered in one "big bang" to the users as phases three and four were developed together which was the core of the system. The plan was fine and according to best practices, but the implementation thereof was not.

iii. Establishing standards

Technology standards were investigated. The system speed and connectivity between branches and Head Office was investigated and tested and proved to be at an acceptable level for the system. However, the system is not at an acceptable speed for the end users. They need a system that is fast to be able to capture a lot of transactions at end of day. With the new system, users spend more time capturing transactions compared to the current system due to the system's connectivity, i.e. users have to wait a while before any feedback gets returned indicating that the transaction was successful. It even takes the user a long time to connect to the logon page.

Development standards were discussed with the developers, but not enforced. Also, the programming standards were not discussed in depth. Naming standards were discussed but standards for database access, etc. were not. Standards, for example, should have been discussed concerning when to use VBScript and when to use JavaScript in web development. This is to ensure that each program throughout the system has the same programming standards so that all developers will be able to maintain each other's codes.

Documentation was discussed in length but during the development, these standards were not enforced. After a while, documentation was not done when changes were being requested due to time constraints. However, documentation should not be ignored as many important facts will be forgotten when post documentation is done. There was also no consistency in where the documentation is stored and in what format. Some opted to used the web and some were in Microsoft Word format. Also, there were several folders on different servers where documentation was stored. This made it difficult to update changes as one does not know which is the latest version and in which folder documentation resides. There were too many folders that referred to the same topic, making it even more difficult to search for documents.

b) Attacking risks

The risk management analysis was performed according to best practices as action plans were formulated for each risk to avoid/minimise such risks from occurring. Some plans were executed while some were not. Once the risk analysis was performed, nothing was done to avoid those risks identified. An important risk that was not identified is user buy-in and participation. No attempt was made to get user support during the analysis and development of the system. Even though branches were not giving their feedback on the Use Cases, there were no steps taken to get the support needed. Once the project was in the analysis phase, risk assessment stopped.

Even though a business case was completed for the project to promote understanding and buy-in from project stakeholders, it was not properly communicated to them. As a result, project owners had their own ideas as to what the system should do. The users did not know why the system was being rewritten and did not know about the future plans of the system. They assumed it was being rewritten due to the current system being old. They did not know that the Internal Audit Department considered the system unstable. Accordingly, there was no motivation for the users to assist in the development of the system.

c) Developing the delivery habit

The delivery habit was initially not instilled amongst the developers, as set deadlines were not enforced. There were unforeseen circumstances for this, for example, stored procedures not working properly or Use Cases not being finalised and changes had to be made, but the project management team did not show a sense of concern as it was still the initial stage of the project. However, when the milestone dates were being reached, pressure was placed on the development team to finish the system. Developers then concentrated on finishing the system and ignored system quality because there were several errors that had to be fixed during user testing.

d) Scope management

The scope of the project was clearly defined in various documents, but was not communicated to the project owners. This caused some project owners to have their own ideas about what the system should be doing. As much as the documents were sent to the project owners, it was evident that there was not much support for the development of the system as some of them were unaware about the project plans. When the system was demonstrated, there was a lot of disappointment due to the differences in expectations and ideas. This highlights the concern that all project owners were not involved in the planning stage of the project and therefore did not understand what was to be delivered. It is evident that project stakeholder expectations were not managed properly.

e) Project management methodology

Project plans and business requirement documents were very descriptive and well defined but were not executed accordingly. The project experienced several problems, which also includes political and business issues. There were deadlines assigned to each iteration and even though the first phase of the project did not meet its deadlines, no exception plan was created. Deadlines were just moved forward, but not the milestones. This resulted in the development time for the last two phases being shortened.

When the project management team realized that deadlines were being missed, more meetings were held with the development team to try to ascertain why this was happening and what could be done so that progress could be made. This step was crucial to the project, as the project would have just continued to miss further deadlines. In this way, management gained some control over the project to ensure that the system would be delivered on time. Even though there were several issues raised as to why the developers could not meet their deadlines, milestones were not rescheduled.

5.1.2 User involvement

The project stakeholders were adequately identified, but there was a lack of communication between them and the project management team. Even though documentation was sent to all project owners, there was no support. There were diversities of interests concerning to the project and the project management team did not clearly communicate the goals of the system and future versions planned. Some project owners expected the whole system to be developed all at once and due to that expectation not being met, spoke very negatively about the system.

The project team decided on using a project owner and branch representative, who were knowledgeable about the current system, to handle user requirements. By doing this, the project team had to liaise only with only a few users. This procedure was done well as the responsibilities of the two representatives were communicated effectively.

However, the project team did not inform users at the start of the project about why the system was being developed. Most branches were satisfied with the current system and thought that it was being rewritten due to the current system being old. Due to the users not being aware of the reasons behind the development of the system, there was no motivation and therefore received no support from the users.

The project manager and business analyst decided to visit the branches whilst the system was being operated in parallel with the old. The purpose of the visit was to explain to the users the reason why the system was being developed and the future plans of the system. This visit also provided an opportunity for users to discuss their problems with the current system. This visit proved to be very effective as users were then more motivated when using the system and assisted the development team with more enthusiasm.

5.2 Analysis

The analysts had a session over two days with branch managers and a key member from each branch. This was not performed very well as there were misunderstandings between users and analysts. It was a good idea to have these sessions, but it proved to be very ineffective. The analysts visited the branches to see how each branch operated and to ensure that all requirements for all branches were met. This proved to be effective as the analysts had a better understanding of the whole process and were able to document user requirements.

Due to the Use Cases being given to the developers to develop as soon as they were finalised, it was important that the functions that were dependent on other functions were done last. This was analysed very well as there were no overlaps in functions during development, i.e. developers did not have to wait for another function to be developed before they could continue with their function.

The developers did not have to waste time waiting for Use Cases, as the Use Cases were being completed at a regular pace. Whilst the project team was testing phase three, phase four iterations were already being developed.

The development of the Use Cases was, however, not done according to Use Case principle. Use Cases should be designed for every business process and not function. In this way, the developer would be able to understand the process instead of just one function of a process. Also, validations would be easier to identify as the link between the functions within the process will be understood and developed accordingly. Reports were treated as a separate entity, instead of being included in the process. This has to be done, in order to understand what sort of information users and managers require from a process and to ensure that the database design is designed to provide that information. There is no quick fix solution if this is not done, as the database design structure would have to be changed. Reports are normally tested by the users which can only be done once the functions are working properly. This means that reports can only be tested towards the latter part of the testing process and therefore major changes cannot be accommodated as it is too late in the SDLC.

5.2.1 User participation

Relevant users were identified to get finalisation of user requirements. However, users were expected to read the Use Cases e-mailed to them. The users were seeing Use Cases for the first time and were not trained to understand them. The Use Cases were very lengthy and were not user-friendly. As a result, it was very tedious for users to read through all the Use Cases as there were several Use Cases for them to read. Due to this, users were not motivated to read the Use Cases and therefore did not provide feedback as they did not have the time to read through them.

The Use Cases also consisted of information that was very technical and could have only been understood by the development team. The users skipped these sections because it was not in a format that they could understand. This was irresponsible as there could have been information that was incorrect in those sections. Users found it very difficult to visualise the system because there were no pictures or workflow diagrams in the Use Cases. There was also no user buy-in which did not assist in this process. The reason for there being no user buy-in was because the users were satisfied with the current system and did not know the purpose of the development of a new system. Accordingly, user requirements were incomplete during analysis.

5.3 Design

There were several changes made during the design phase, as user requirements received were incomplete, even though the Use Cases were signed off. This had caused developers to constantly change their codes. Sometimes the developers were not informed about the database changes and as a result, the function that used to work previously did not work during testing. This highlights that there was no communication between the database administrators and developers.

Developers were not able to write the stored procedures for their functions and therefore, most of the time they did not have anything to do as they were waiting for the completion of their stored procedures. Eventually one DBA took over writing all the stored procedures.

Due to time constraints in the last phase, the developers did not have the time to test their own codes before it was being tested in QA. This caused several errors like buttons or functions not working. The developers should not have sacrificed quality even if there were time constraints. If the quality of the system is ignored, users are not going to trust the new system.

Also, testing by the project team should not have been ignored. The project management team should have revised project milestones to ensure that when the users were testing, they were testing only for functional faults, and not for errors.

5.3.1 Multiplicity of user interfaces

The system was to be used by the financial institution's Head Office and branches. Therefore it was easy to stipulate what the minimum standards should be. The project management team knew that the branch's clients for the next version will also use the system, and accordingly investigated their hardware and software as well.

Users in the branches were using Microsoft Windows 95 operating system with Internet Explorer (IE) 5.5. However, the developers used Microsoft 2000 operating system with IE 6. This created some problems as the system was designed without checking if the system would work on an earlier version of IE. This resulted in some functions not working on user machines but working on the developer's machine and this caused the developer to recode the program to allow it to work on the earlier version of IE which involved time being wasted.

The minimum screen resolution standard for the system is 800 by 600 pixels. It was later discovered that when users were using a higher screen resolution, i.e. 1024 by 768 pixels, there was a lot of space on the screen. Users then wondered why the developers could not increase the capture screen to allow them to capture more transactions without having to scroll down.

5.3.2 User interface design principles

The overall explanation of the system was not discussed with the developers. Due to this, developers could not be very creative in their design, as they just had to follow the Use Case. With the Use Case, the developer could not relate the function to the whole system, and therefore did not understand the purpose of the function. This made validity checks very difficult to determine. Only the basic validity checks were programmed according to the Use Case. Also, due to the Use Cases not being process driven, developers could not understand the process and therefore designed screens without considering ease of use.

Numerous reports were not designed according to user requirements. Reports were not displayed in a way that was useful to the user. Even though developers had the structure of the report, due to them not understanding the system or process, they designed the report according to what was shown on the existing reports. If a user requested a report of transactions on a specific date, for example, the report worked, but if the user requested a report for a period, it did not work as this information was not provided to the developers.

It was discovered during user testing that users were not proficient with Windows operating systems. Due to the old system being a mainframe application, they were not used to drop down lists etc. They previously used use just the keyboard to capture information. With the new system, they were required to use both the mouse and the keyboard, which slowed the data capturing.

The menu system and system architecture was very well designed as the users found the system easy to use. The menus are very descriptive and it is very easy to navigate through the system. The TASK LIST was a major improvement to the current cash management system as it made it easier for the user to keep track of the transactions still awaiting verification and authorisations.

5.3.3 Criteria for developing a web site

The system met the information quality required for web systems. The information was transmitted through a secure line and information is presented in ways that are understood by the users. The system allowed for easy navigation. There was also a help facility, which explains in detail how each function works, and how the system operates. This help facility was readily accessible and interactive as there are step-by-step procedures available.

Users were, however, not very enthusiastic about using the system, as it took long for the logon page to appear and once they logged on successfully, the main screen took a while to load. Users also found that it takes a while for information to appear on the screen. Graphics, and the number of database updates, are normally the reasons for the slow responses. However, some graphics were cached while others were built in dynamically. The database connections were kept to a minimum. Hence, the reason for the slow response was due to the network connectivity to Head Office and is not a result of design flaw. The stored procedures and program codes were optimal.

5.4 Implementation and support

The users were asked to test the system according to the phases that the project was split into. There were a lot of support personnel to assist with queries from the users. Further a procedure was in place for users to report any errors and/or functional faults.

The project team ignored the QA testing due to time constraints for the last phase. This caused the users to pick up the errors as well as functional faults. Due to the errors, users were delayed with their testing as they could not progress until the errors were rectified. Due to these errors, negativity was stirred up amongst the end users. Users did not trust the new system as they found it to be unreliable.

5.4.1 Software change management

Documentation of new requests and changes were not done timeously due to time constraints. However, a procedure should have been in place to store these new requests properly.

Errors and functional faults that were reported were not documented properly. Due to this, users were unaware of all the errors and functional faults reported. As a result, users were reporting errors and faults that were already reported by other users. The status of all faults reported was not communicated to users, i.e. users did not know when the problem was going to be fixed and were therefore making enquiries continuously.

There was no change management in place. As soon as users started requesting changes, and where possible, it was done without consulting the project management team. The Use Cases were not updated which resulted in them being outdated.

5.4.2 Resistance to change

There was enormous resistance towards the new system. This is normal for any new system that is implemented. Users eventually get used to the new system and the way things should be done and ultimately this resistance is overcome. Means to reduce resistance were not done. The users were not involved early in the development and there was no communication to the users as to why the system was being developed. Further, when the system was introduced to the users, it contained several errors and did not meet their requirements which created a lot of negativity. The users still prefer the current system as users feel the system takes too long to capture transactions and to get confirmation of transactions captured.

5.5 Conclusion

From the evaluation of the system, it is evident that the project team did not perform according to best practices in developing an information system. The project therefore experienced several problems and was mainly due to there being no user buy-in. A lack of communication existed between project stakeholders and the project team and therefore there were incomplete user requirements. Expectations were not managed. These problems could have been avoided. Recommendations are provided in the following chapter to improve the information systems development process.

CHAPTER SIX: RECOMMENDATIONS

6.1 Introduction

The aim of this research was to evaluate the development of the cash management system developed by a financial institution for its Head Office users and branches. The project experienced several problems. The research aimed to investigate and identify best practices for developing such a system are. The objective was to ascertain what was and what was not followed by the project team according to best practices.

In order to establish a theoretical framework for this research a literature review was conducted to determine best practices for developing an information system. The review was conducted using the work of recognised and reputable academics who are leaders in the field of information systems development.

Interviews were conducted with key people affected by the system in order to determine what steps were taken to develop the system in the various phases of the systems development life cycle. The interview also gave people an opportunity to express their opinion as to what they felt went wrong with the system and to make recommendations about how they perceive the system should have been developed based on the mistakes experienced.

Conclusions and recommendations are drawn using the findings of the research study together with the literature review. The recommendations will be presented in terms of the areas that require change to improve the development of information systems in the financial institution.

6.2 Recommendations

It will be necessary to summarise the findings and consolidate the results of the study before making recommendations.

6.2.1 Risk management

- **Findings:**

Risk analysis was done and documented. However, these risks were not taken seriously as no proactive actions were taken to avoid or minimise the risk from occurring.

- **Recommendations:**

Risk assessment should be an ongoing process throughout all the stages of the development of the project and not just in the planning stage. Action plans as to how the risks can be reduced or avoided must be discussed and most importantly, implemented. Once this is finalised, the status of the risks should be analysed at every progress meeting and the project team should try and identify or pre-empt more risks throughout the development of the project.

Risk analysis should be taken very seriously especially by management in order to ensure project control.

6.2.2 Project and management control

- **Findings:**

The initial phases of the project missed set deadlines and milestones were moved not forward.

- **Recommendations:**

The project management team should have revised the final project milestone as soon as they were informed that the project was experiencing problems. With the initial phases going over schedule, it reduces the time allocated for development during future phases. It is better to deliver a working system free of errors, than a system that has not been tested properly due to time constraints. In this way, users can trust a reliable system. When milestones are set, it is very difficult to predict the problems that will be experienced. Therefore, these milestones should be revised on a regular basis to ensure that milestones are met and a system that is of high quality can be delivered to the users.

6.2.3 Project stakeholder buy-in and support

- **Findings:**

There was a lack of buy-in and support from project stakeholders. Each project stakeholder had his/her own idea about what the system should do and some did not even understand why the system was being developed. User buy-in and support is one of the most essential parts of developing an information system. Without user buy-in and support, the project team will have problems receiving user requirements and will encounter resistance to change.

➢ **Recommendations:**

A serious effort should be made to encourage project stakeholder buy-in and support. Regular meetings should be held to keep project owners up to date with the project. This will also assist in keeping them informed about the goals of the projects. More demonstrations of the system should be offered to solicit project owners involvement in the system and to keep their expectations in line with those of the project team.

End users should be informed at project initiation about the background of the project and should include the reasons for the development of the system and its future plans. If these plans were going to benefit the user by making their job easier, their attitudes towards the system would improve and they would be motivated to assist in providing user requirements.

6.2.4 User participation

➢ **Findings:**

Users did not provide feedback when given Use Cases to read and validate. This was due to Use Cases not being user-friendly. The Use Cases were too lengthy and tedious to read.

➢ **Recommendations:**

The Use Case should be in a format that is understandable by the user. All computer jargon should be avoided in the Use Case provided to the user. More workflow diagrams should be provided so that users could understand and be able to visualise the process. Use Cases should also not be too lengthy as users are not keen to read through each Use Case. If a new tool is used, it is always beneficial to teach the user about how to use the tool instead of letting them learn on their own.

6.2.5 Database design

- **Findings:**

Several database changes were made during the development of the system caused numerous programs to be rewritten or repaired. There was also little communication between the DBAs and developers.

- **Recommendations:**

The database should be finalised before developers start programming. This can only be done once the whole system is analysed properly. The analyst, the representatives nominated and DBA should sign off Use Cases before developers start coding. Due to the preliminary design of the database, it was later found that the database design did not meet the requirements of some functions. There should also be a formal communication structure in place to ensure that all affected parties are kept informed about the changes.

6.2.6 Development structure

- **Findings:**

The presentation of results shows that the system is designed using a three-tier architecture in which stored procedures is the middle layer between the database and the front-end. The stored procedures were written by the DBAs which frustrated the developers as they became just front-end developers. Developers had to wait for stored procedures to be completed before they could continue with their work or when stored procedures were changed, they were not informed.

➢ **Recommendations:**

The project team would have to look at the feasibility of allowing developers to write their own stored procedures. In this way, they would be able to understand the process better, and be able to debug easier. This will ensure that developers will not be sitting idle waiting to continue with their work. This process will also make developers fully responsible for their functions and thereby allowing them to take full responsibility in ensuring that the deadlines are met. By doing this, there will be more technical people who understand the system design instead of just one.

One factor that needs to be considered is that the developers have not written stored procedures before, and to ensure that database access and updating of information is optimal, there should be a QA environment to test the stored procedures.

6.2.7 Maintainable programs

➢ **Findings:**

The results of the study showed a lack of standards in place when programming the system. This compromised the ability of the development team, as they were unable to maintain each other's code.

➢ **Recommendations:**

Standards were not followed due to the different levels of skill of the developers, however standards should be enforced. If the levels of skills of developers in the development team are different, procedures should be in place for developers to discuss and conform to a development style.

Developers with lower levels of skills, should be trained or if there is no time, to at least be shown what has been done. If a programming style cannot be determined due to the diversity of functions regular meetings should be held with the developers to be able to discuss how the functions were developed. This would uphold the quality of the system, as developers can provide suggestions to improve on the codes, screen design, etc.

6.2.8 Development design plan

- **Findings:**

The results of the study show that the core of the system was developed in a shorter time than it did to develop the administrative functions. As a result, the core of the system was not tested properly and several functional faults were discovered during the user testing.

- **Recommendations:**

With a large project, it is recommended that focus should mainly be on the core of the system. In this way, more time can be allocated to the main business processes to develop and to be tested thoroughly before users start to test. There would be more time available to fix any functional faults that are reported.

Developers would find administrative functions easier to develop than core business functions as they would understand those functions easier. Testing administrative functions should be easier as they are relatively straightforward.

6.2.9 Scope management

- **Findings:**

Users requested several changes during the user testing and parallel runs. There was no procedures in place to manage these changes.

- **Recommendations:**

Each user request that is not according to the Use Case should be discussed first with the project team before making any changes to the programs. This will also ensure that documentation is completed. The project team should then decide if this is within the scope of the project or whether the request should be done at a later stage.

6.2.10 Documentation

- **Findings:**

Documentation was ignored during the design and implementation phases of the project.

- **Recommendations:**

Procedures should be in place when users request changes. Steps should be followed, beginning with an update of the Use Case prior to making any changes to the programs. Also, documentation should have source control to ensure that previous versions are stored for future reference. This avoids the Use Case being untidy with vast amounts of information being literally struck off on the document.

6.3 Conclusion

The previously mentioned recommendations indicate that it is possible for the project team to improve the development process of information systems. Even if internal politics do exist, there are ways to work around those politics if procedures are in place. Risk analysis is a very important step in the project and steps should be taken to minimize risks or avoid them completely.

From this research it is also evident how important it is to have user buy-in and support at all stages of the project. It is also important to manage project stakeholder expectations in order to ensure that their expectations are in line with those of the project team.

BIBLIOGRAPHY

Allen, C.P. (1991) Effective Structured Techniques from Strategy to Case, 1st Edition, Prentice Hall, Britain.

Ary, D., Jacobs, L.C & Razavieh, A. (1979) Introduction to research in education, Holt, Rinehart & Winston, New York.

Bocij, P., Chaffey, D., Greasley, A. & Hickie, S. (1999) Business Information Systems – Technology, Development and Management, 1st Edition, Pitman Publishing, Great Britain.

Boyette, D. (2002) Gathering User Requirements with Care, TechRepublic [Internet], 21st March, Available from: <http://www.techrepublic.com/article.jthtml?id=r00720020319gan01.htm> [Accessed 7th May, 2003].

Bredemeyer (2000) Fundamental Requirements and Use Cases, Bredemeyer Consulting [Internet], 27th June, Available from: < http://www.bredemeyer.com/use_cases.htm> [Accessed May 8th, 2003].

Browne, G.J. & Ramesh, V. (2002) Improving information requirements determinations: a cognitive perspective, Information & Management Journal, 39 (8) September, pp. 625-642.

Cardozo, E.L. (2002) The Seven Habits of Highly Effective Iterative Development, Rational Software.

Connell, C. (2002) Best Practices for Web Strategies in 2003, Vignette Corporation, September.

Cooper, D.R. & Schindler, P.S. (2001) Business Research Methods, 7th Edition, McGraw Hill, Singapore.

Cronbach, L.J., & Suppes, P. (eds.) (1969) Research for tomorrow's schools: disciplined inquiry for education, Macmillan Publishing, New York.

Daum, B. & Horak, C. (2001) The XML Shockwave, 1st Edition, Software AG, Germany.

Eva, M. (2001) Requirements acquisition for rapid applications development, Information & Management Journal, 39 (2) December, pp.101-107.

Gay, L.R. (1987) Education Research, 1st edition, Merrill Publishing Co., Columbus.

Hartman, F. & Ashrafi, R.A. (2000) Project Management in the Information Systems and Information Technologies Industries, Project Management Journal, 33 (3) September, pp.5-13.

Jain, A. (2002) Application Development: The Next Hurdle, White Paper, Syntel, August.

Jay, K.E. & Smith, D.C. (1996) A Generic Change model for the effective implementation of information system, South African Journal of Business Management, 27 (3) September, pp.65-70.

Jiang, J.J., Chen, E. & Klein, G. (2002) The Importance of Building a Foundation for User Involvement in Information System Projects, Project Management Journal, 22 (1) March, pp.20-25.

Lauter, S. (2002) Project Management – knowing how to fail helps you succeed, Network Times, 13 (9) October, p.6.

Lawrence, E., Netwon, S., Corbitt, B., Braithwaite, R. & Parker, C. (2002) Technology of Internet Business, 1st edition, John Wiley & Sons Australia Ltd, Singapore.

LeClair, D. (2002) Managing eBusiness Development, White Paper, 11 April, Computer Associates.

Levitt, J. & Harbaugh, L. (2000) Three Keys for Building a Super-Fast Website, InformationWeek [Internet], 5th June, Available from: <http://www.informationweek.com/789/we2.htm> [Accessed May 7th, 2003].

Lientz, B.P. & Rea, K.P. (1999) Breakthrough Technology Project Management, 1st Edition, Academic Press, California.

Lin W.T. & Shao, B.B.M. (2000) The relationship between user participation and system success: a simultaneous approach, Information & Management Journal, 37 (6) January, pp.283-293.

Liu C. & Arnett, K.P. (2000) Exploring the factors associated with Web site success in the context of electronic commerce, Information & Management Journal, 38 (1) October, pp.23-31.

Munro, M.C., Huff, S.L., Marcolin, B.L. & Compeau, D.R. (1997) Understanding and Measuring User Competence, Project Management Journal, 33 (1) November, pp.45-55.

Raveendra, V.V.S. (2001) E-Buiness Application Development: The Paradigm Shift from In-House Application Development, First Monday - Peer-Reviewed Journal on the Internet [Internet], 1st January, 6 (1), Available from: <http://www.firstmonday.dk/issues/issue6_1/raveendra/> [Accessed April 30th, 2003].

Roode, J.D. & Smith, A.J. (1989) User involvement in systems development, South African Journal of Economic and Management Sciences, Issue 2, November, pp.7-20.

Schumacher, S. & McMillan, J.H. (1993) Research education: a conceptual introduction, Harper Collins College Publishers, New York.

Slavin, R.E. (1984) Research methods in education. 1st edition,Prentice-Hall, Englewood Cliffs.

Smart, K.L & Whiting, M.E. (2001) Designing systems that support learning and use: a customer-centered approach, Information & Management Journal, 39 (3) 20th December, pp.177-189.

Sodbinow, E.S. (2001) The Customer Experience Begins at the Beginning, White Paper, Patricia Seybold Group.

Sowell, E.J. & Casey, R.J. (1982) Research methods in education, 1st Edition, Wordsworth Publishing, Belmont.

Van den Aardweg, E.M. & Van den Aardweg, E.D. (1988) Dictionary of empirical education, E & E Enterprises, Pretoria.

Whitten, J.L., Bentley, L.D & Barlow, V.M. (1994) Systems Analysis and Design Methods, 3rd Edition, Irwin Inc, USA.

Wiersma, W. (1991) Research methods in education, 1st edition, Allyn & Bacon, Boston.

Yeh, Q. & Tsai, C. (2001) Two conflict potentials during IS development, Information & Management Journal, 39 (2) December, pp.135-147.

APPENDIX A

Interview with Project Manager and Project Leaders:

1. Whose idea was it to start the project?
2. Who made the milestones and deadlines?
3. Was a risk analysis done?
4. If so, were actions in place to minimise or avoid them from occurring?
5. Were project stakeholders identified?
6. If so, what plans were made to get them involved in the project?
7. If not, was there any reason why they were not identified?
8. When did the project stakeholders get involved with the project?
9. Did you get user buy-in and support?
10. If not, were there any steps taken to get their buy-in and support?
11. What development tools did the project management team decide upon?
12. Were those tools used and followed by the rest of the team?
13. Were programming standards discussed with the developers?
14. Were there documentation standards discussed?
15. Were deadlines set for each phase and each iteration?
16. Was there a procedure that was followed should deadlines not be met?
17. If not, what was done when deadlines were being missed?
18. Was there any change management in the project?
19. In your opinion, why did the project experience problems?

APPENDIX B

Interview with Business Analyst

1. How did the analysis for the project start?
2. Was this the first time that the business analyst team used Use Cases?
3. When did the database design get finalised?
4. When did the developers get involved?
5. Were the Use Cases done only
6. Were the project owner and branch representatives asked to sign off the Use Cases?
7. Were changes then requested after the sign off? If so why?
8. Were the Use Cases updated immediately? If so, was there any source control on the Use Cases to keep track of the changes?
9. When receiving changes after sign off, was there any sort of change management that you had to follow?
10. How was the development team informed about the changes?
11. Did the users provide feedback when asked to read the Use Cases?
12. How were users informed of the changes that were being requested (i.e. requests from all branches)?
13. In your opinion, why did the project experience problems?

APPENDIX C

Interview with Database Administrator

1. When was the database designed (i.e. during Analysis or after sign off from Use Cases)?
2. Were several changes made after the design was complete? If so, were there major changes?
3. Were there any change management procedures?
4. Why were the developers not allowed to write stored procedures?
5. Initially, two other DBAs were involved in writing stored procedures, then just one. Why did this happen?
6. When changes were made to the stored procedures and database, how did you inform the developers?
7. In your opinion, why do you think the project experienced problems?

APPENDIX D

Interview with Developers

1. When did you get involved in the project?
2. Were programming standards discussed to you? If so, were they followed?
3. Can you maintain another developer's code?
4. Did you ever use Use Cases prior to this project?
5. Did you understand the overall goals of the system whilst developing?
6. Did you find the Use Cases beneficial?
7. Was the database design discussed with you?
8. Was each Use Case explained to you prior to developing?
9. Were you given business processes or functions to develop?
10. Did you understand the business processes or functions given to you?
11. Did you understand how the business process or function fitted in with the rest of the system?
12. Were there changes made to Use Cases after it being given to you?
13. If so, did this impact on your development?
14. Were you informed about changes made to the database design and stored procedures?
15. Were there a lot of changes requested for administrative functions?
16. Were there a lot of changes requested for the core functions of the system?
17. Were there a lot of changes requested for the reports?
18. Users reported a lot of errors during the testing? Any reasons why this happened?
19. Did you update documentation when changes were being made?
20. In your opinion, why did the project experience problems?

APPENDIX E

Interview with Users

1. When did you get to know about the project?
2. Was there any communication, verbally or written, to explain why the project was initiated and the future plans of the system?
3. Did you have any knowledge of Use Cases prior to the project? If not, was there any training provided?
4. Did you find the Use Cases easy to learn and read?
5. Did you understand the purpose of the Use Cases?
6. Were you eager to assist in providing requirements to the project team?
7. During user testing, did you find that the system had errors?
8. If so, did the errors occur on all, some or a few functions?
9. During user testing, did you find that the system have functional faults, i.e. the system did not do what you wanted it to do in order to complete your work?
10. If so, did the functional faults occur on all, some or a few functions?
11. Was there any procedure in place to report errors or new requests?
12. Were you aware of errors and functional faults reported by the other branches?
13. What was the attitude amongst users – positive or negative?
14. Did you know the status of these change requests since submission?
15. Do you feel that the system is easy to learn and use?
16. Is the capturing of information easy?
17. Did you have any problems getting support from the support personnel?