# The Effect of Requirements Engineering on the Success of System Implementation: A Comparative Case Study

A Dissertation Submitted in Partial Fulfilment of the Degree of

MAGISTER INGENERIAE

in

ENGINEERING MANAGEMENT

at the

FACULTY OF ENGINEERING AND THE BUILT ENVIRONMENT

of the

UNIVERSITY of JOHANNESBURG


by

Annlizé Marnewick

October 2010

SUPERVISOR: PROF. JHC PRETORIUS

CO-SUPERVISOR: PROF. L PRETORIUS

# Abstract

Requirements engineering forms an integral part of software engineering. The purpose of requirements engineering is to provide high quality requirements for a system or solution. These requirements are then utilised by developers to produce a high quality system. They also assist project managers to better plan the schedule and costing of information technology projects, resulting in cost savings.

The problem is that although formal definitions and processes do exist for requirements engineering, projects are still failing due to the poor quality of requirements. This study investigates this phenomenon, in particular to understand why project teams cannot deliver high quality requirements. This is done against the background of the processes and standards available to organisations. The root cause of the problem is researched to determine whether the processes are the cause or whether other factors are contributing to poor quality requirements.

This study makes use of two cases within one organisation to determine what the contributing factors are with regard to poor and good quality requirements. The cases provide information on why one project delivered good quality requirements and another project within same organisation, the same business unit, with the same support structure, delivered poor quality requirements. It is perceived that the case study method was a valid method in this particular research study as it provided the researcher with in-depth knowledge and observations on how organisations deal with the process of requirements engineering.

It was found that the quality and clarity of communication or the lack thereof plays a significant role in the quality of requirements. This research provides an alternative view on the factors contributing towards poor quality requirements. This implies that organisations can train or educate requirements engineers in communication skills. The skill of communication allows a requirements engineer to create a trust relationship with customers, and this empowers him/her to elicit good quality requirements from the users.

## ACKNOWLEDGEMENTS

Thanks to my Father in Heaven for the discipline and insight to do this research.

My appreciation to all former colleagues that were prepared to participate honestly in questionnaires.

My supervisor and co-supervisor for the guidance they provided.

Carl for all the critical reviews.

## CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS

| | |
|---|---|
| CAQDAS | Computer-assisted qualitative data analysis |
| IEEE | Institute of Electrical and Electronics Engineers |
| IIBA | International Institute of Business Analysis |
| RML | Requirements Modelling Language |
| SADT | Structured analysis and design technique |
| SAGE | Semi-automatic Ground Environment |
| SWEBOK | Software Engineering Body of Knowledge |

# CHAPTER 1: INTRODUCTION

Requirements engineering is the process of activities, elicitation, analysis, specification and validation performed to understand a problem that needs a software system as a proposed solution. Software projects are known for their high failure rate. This high failure rate of software projects have not improved over the years as requirements engineering matured (Maiden, 2008).

Although numerous methodologies, tools and techniques are available to address the requirements, the winning combination to ensure success in projects has not yet been found. This phenomenon is confirmed by a recent study that found that one in eight software projects is truly successful (McManus and Wood-Harper, 2007). This study also highlighted that the lack of good requirements is always on the list as a key contributing factor towards failed projects (McManus and Wood-Harper, 2007).

## 1.1 Requirements Engineering History

A software development process to be used in the Semi-automatic Ground Environment (SAGE) system was created using the same approach as that used to develop hardware in 1956. This process was a top-down one, and formulating requirements was one of the steps. It became apparent that software is different from hardware, modifications were much easier than in hardware, and so it soon followed a "code and fix" approach compared to the thorough hardware design reviews before production commenced (Boehm, 2006). The maintenance process of software was also different from that of hardware: software did not wear and tear like hardware. However, software was more people intensive than hardware and soon the demand for software surpassed the supply of engineers. Software development positions were taken up by creative people, which often resulted in spaghetti code, when fixes led to patched code (Boehm, 2006).

During 1968, it was acknowledged that more structured approaches were required for software engineering (Randell and Naur, 1968). A model was introduced, called the Waterfall model for software engineering. With this model, software was developed in a sequence of activities where formulating requirements was the initial step (Royce, 1970).

With no formal techniques in place, requirements were written in a natural language. The empirical study of Bell and Thayer (1976) provided evidence that errors in the requirements had a significant impact on the quality of the software developed from these requirements. A need for techniques to improve requirements was identified, which established requirements engineering as a subfield of software engineering in the early 1970s (Greenspan et al., 1994).

In the late 1970s, formal requirement techniques were introduced which utilised structured analysis methods such as the structured analysis and design technique (SADT) of (Ross and Schoman, 1977). This method (i) defined what requirements engineering should be doing, (ii) defined the techniques required to do requirements engineering and (iii) introduced the use of graphical techniques to generate specifications that facilitate communication between all parties involved in the process. Similarly, DeMarco (1979) and Gane (1979) used

dataflow diagrams, data dictionaries, structured English decision tables and decision trees to generate requirements.

Typically the starting point of these techniques was to establish system functions. An alternative starting point was suggested to (i) create a model of the reality in which the system will operate and (ii) then consider the detail functions required (Jackson, 1983).

The shortcomings of the structured analysis methods were as follows (Maiden, 2008):

- The requirements included no reasoning why the solution was needed. They focused purely on the functional decomposition using graphical methods.
- No considerations were given to requirements such as performance, security and reliability, which are typically classified as non-functional requirements.
- These models were not self-explanatory and therefore could not be used as communication to stakeholders.

In the 1980s, the focus moved to represent the knowledge accumulated during the requirements engineering process and not just the requirements. Greenspan et al. (1994) developed Requirements Modelling Language (RML), which supported an object-oriented approach.

Ethnography techniques were introduced in the 1990s to understand the system environment better. This approach focused on users and user interactions with systems, rather than the data itself, its structure and processing (Sommerville et al., 1992). During the 1990s, it was also acknowledged that users did not necessarily know their requirements and that requirements engineering is a discovery process (Maiden, 2008).



Terminology changes were made. The requirements process was initially known as 'engineering' and changed later to 'requirements engineering'; currently 'requirements engineering and management' is used. The requirements produced also changed a few times and are mostly known as customer requirements specifications (Hood et al., 2008). The terminology changes are shown in Figure 1. For the purposes of this study the term "requirements engineering" will be used.

| | Requirements Process | Requirements | |
|---|---|---|---|
| 1970's | Engineering | Customer Requirements Specification | Product Specification |
| 1980's | Engineering | User Requirements Specification | Functional Requirement Specification |
| 1990's | Requirements Management | Stakeholder Requirements Specification | System Requirement Specification |
| 2000's | Requirements Management & Engineering | Customer Requirements Specification | System Requirement Specification |

**Figure 1: Timeline of terminology change (Hood et al., 2008)**

From this timeline it is evident that researchers have made significant progress in tools and techniques to improve on the quality of requirements. The research world and the practice of requirements engineering are two worlds apart (Siddiqi, 1996). The tools and techniques are not rigidly applied in practice; in many instances the requirements engineering discipline is not understood or seen as the capturing of a user requirement and moving the captured information to a development team. This is confirmed by researchers such as Sutcliff et al. (1999), Viller et al. (1999) and Walia and Carver (2009), who state that many requirements are still produced in practice with errors which then leads to problems during system implementation. With the history of requirements engineering in mind, what is currently happening in practice is investigated in the following section.

## 1.2 Problem Statement

Software project failure is unacceptable for any organisation in the economic system. Failure is often defined in research as projects that do not meet the time, cost and quality criteria. Quality is therefore one of the main reasons why projects fail when the project deliverable does not function as required; the customer is not satisfied and then finds an alternative (Agarwal and Rathod, 2006). Reasons noted by researchers why projects fail include the following:

- Unstable i.e. changing requirements are one of the main reasons why runaway projects are unstable (Glass, 2003). Runaway projects are where schedule, cost, or functionality original estimates go out of control.
- Requirements and scope changes after a project has been initiated are primary reasons for project cancellation (El Emam and Koru, 2008).
- Delivery decisions are made without a complete set of requirements (Verner et al., 2008).



Hooks and Farry (2001) believe that if the project starts with good requirements, success will be achieved in terms of quality, cost and schedule. They note that rework will be eliminated, unnecessary features rooted in poor requirements will be excluded and the fit between the solution and the customer's needs will be better.

Requirements are listed by all of the above authors as a main contributor to project failure or success. The problem statement of the study therefore is:
**Poor quality requirements, as often found in requirements engineering, are responsible for project and system implementation failures, and exponential cost.**

The next section focuses on whether the failure to produce quality requirements currently has an impact in industry on project success.

## 1.3 Requirements Engineering's Contribution to Project Success or Failure

Requirements engineering plays a key role in determining the success or failure of projects, as well as the quality of the systems delivered (Kamata and Tamai, 2007; Damian and Chisan, 2006). Poor requirements engineering is identified as one of the main contributors to the failure of system implementation (Hofmann and Lehner, 2001).

Recent industry reports by IAG Consulting have confirmed the impact of requirements on projects. The IAG Business Analysis Benchmark report in 2008 (Ellis, 2008) analysed the impact of poor requirements on organisations:

- Organisations with poor requirements and analysis capability have a ratio of three project failures for every one project success.
- Forty per cent of the information technology development budget for software, staff and external professional services will be consumed by poor requirements in average organisations using average analysts.

A concern about these reports is that they are based on managers' subjective perceptions and not supported by scientific data. To address this issue, Kamata and Tamai (2007) provided findings to support and prove that there is a correlation between the quality of requirements and the success of projects.

Another local study was done to look at project success in South Africa (Labuschagne and Marnewick, 2009) and the two main factors mentioned as the reason for the failure of complex projects were:

- People (communication between project team members)
- Direction (clarity of business objectives; clarity of requirement definition)

From the evidence provided, it is clear that academia and the business research community, both internationally and locally in South Africa, seem to be in agreement that requirements have a significant role to play in project success or failure. The researcher believes that by focusing this research on requirements engineering, a contribution can be made to the requirements community as discussed in the next section.



## 1.4 Research Objectives

The research objectives were to:

- Identify the factors that contributed to quality requirements during the requirements engineering process of a project.
- Consider the impact of quality requirements on project success or failure.

This study result identifies the factors contributing to delivery of quality requirements and how these affect project success. This will add to the existing body of knowledge within the requirements engineering community.

## 1.5 Research Questions

To explore if requirements delivered during the requirements engineering process contribute to project success or failure, answers to the following questions were investigated:

- **RQ 1:** Why do some implementation teams produce good quality requirements and others poor quality requirements?
- **RQ 2:** How does the quality of requirements contribute to project success?

If the root causes of good or poor quality requirements can be identified, solutions can be applied and tested in an attempt to reduce the number of failed projects in practice. The research process followed to answer the research questions is explained next.

## 1.6 Research Process

A structured approach as adapted from Cooper and Schindler (2008) was followed to answer the researched questions and is illustrated in Figure 2.

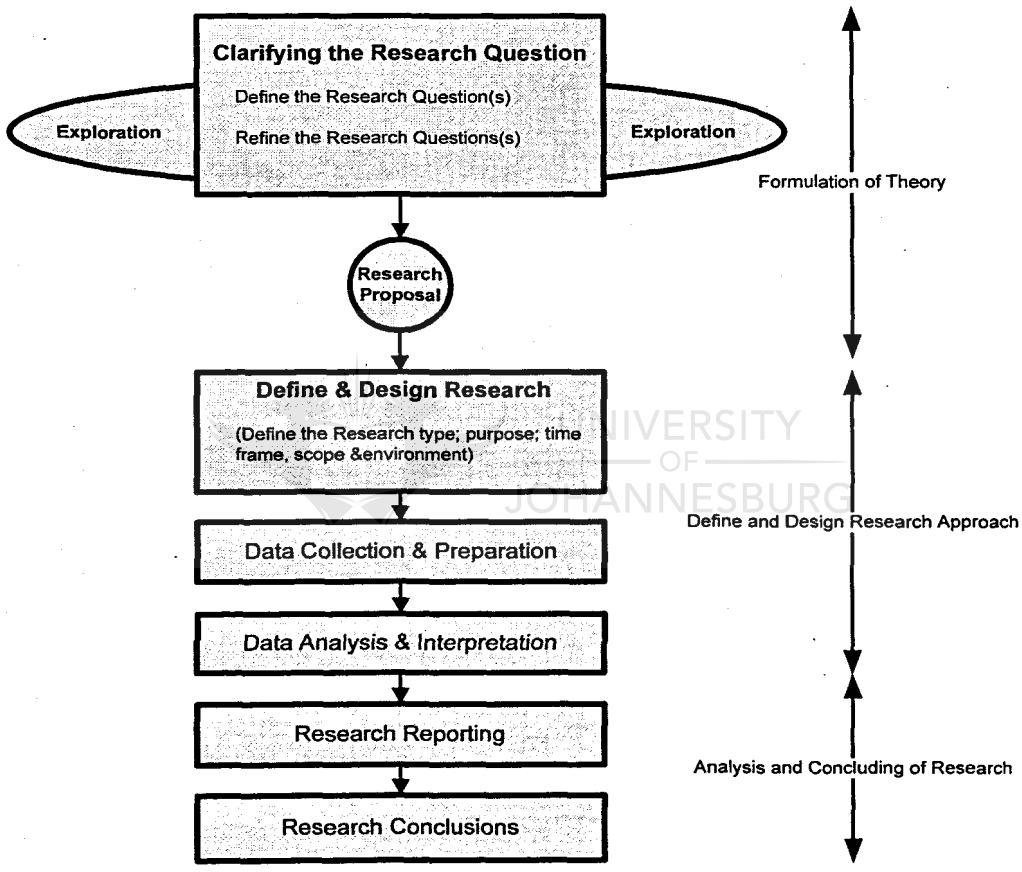


**Figure 2: Research process (Cooper and Schindler, 2008)**

The researcher applied this structured approach during the research process as described below.

### 1.6.1 Clarification of Research Question

The starting point for research is to identify a dilemma faced in practice (Cooper and Schindler, 2008). Prior to commencing the research, the researcher was an active resource in industry, implementing projects. The belief that requirements engineering influences project success or failure was the trigger to explore what other

researchers have found regarding requirements engineering challenges. With this as the point of departure, published studies and industry reports on requirements engineering were explored to define the problem area to be researched (Olivier, 2009).

### 1.6.2 Research Proposal

A written research proposal was produced, the main purpose of which was to:

- Obtain approval from the academic institution concerned for research.
- Present the problem area faced in practice to be researched, including the importance of the problem.
- Suggest data to be used to answer the research questions.
- Present a plan of how the research was to be executed.

### 1.6.3 Definition and design of research

In conducting research, a plan is required on how the research objectives will be research and questions answered (Cooper and Schindler, 2008). In this case a problem-solving process was applied during the define and design step, as discussed in (Olivier, 2009).

- An understanding of what other researchers in this field have previously established was developed by conducting a literature study. This assisted in formulating theory.
- Potential research methods were investigated and the most appropriate method to answer the research questions was selected.
- Data required was defined based on the research method selected to gain an understanding of how the requirements engineering process is executed in practice. The purpose of this was to determine whether there was any correlation between research findings in the literature study and actual real-life scenarios.



### 1.6.4 Data collection and preparation

The research method selected dictated what data would be collected and how (Cooper and Schindler, 2008).

### 1.6.5 Data analysis and interpretation

Data analysis involved the data collected being reduced, summarised into a usable format and patterns in data being identified (Cooper and Schindler, 2008). From this analysis answers to the research questions were derived. It is at this stage that the degree of consistency of results with the theory was determined.

### 1.6.6 Research reporting and conclusions

Using the evidence collected and interpretations made, research findings were documented. Answers to research questions found were evaluated. Proposals were made on how answers to research questions fit into the current body of knowledge of the requirements engineering community.

The research process steps are presented in various chapters of this report. The following section provides an overview of where each step is discussed.

## 1.7 Research Report Layout

Chapter 2 is a literature study that investigates what other researchers have already found in the requirements engineering community. The purpose of the literature study was to survey all relevant information that has previously been published (Olivier, 2009). The scope of this literature study was the requirements engineering process, including its definition and how it fits into software engineering. Furthermore, the key challenges faced in requirements engineering that affect quality requirements were investigated, as well as how the quality of requirements contribute to the project success.

In chapter 3 alternative research methods are first evaluated and a motivation why a case study was selected as the most appropriate research method to obtain answers to the research questions is given. Secondly, the research design is explained. The research design details what questions the study answers, identifies the data that are relevant to answer these questions and explains how relevant data were collected and analysed (Cooper and Schindler, 2008) (Yin, 2009).

Chapter 4 provides background on the two projects selected for the case studies, as well as a description of how the case study evidence was collected. It also details how the evidence was analysed according to a rigid systematic approach that was followed to ensure a solid research design.



The data patterns derived during the analysis for each individual case study are discussed in chapter 5. A cross-case study analysis is done of the two cases in order to categorise all similarities and differences. This validates whether findings can be replicated across the different cases.

Chapter 6 concludes this study.

## 1.8 Conclusion

Progress has been made over the past 60 years to develop the field of requirements engineering, but the practical application of this field is still noted by research as a main contributor to failed projects. In general, all projects are developed through a certain process and for the final delivery of the project, thorough input is required at each stage of the project. It is therefore vital that the first stage, which is requirements engineering, be done accurately. If not, all other stages or processes can be done perfectly but the project will still fail due to poor quality requirements in the initial stage. In practice, requirements engineers would benefit by understanding what the contributing factors are to quality requirements. This will enable them to do requirements engineering right the first time.

Chapter 2 investigates whether the contributing factors to quality requirements are known. The current body of knowledge can be expanded through the development of methods or techniques that are suitable and

applicable in practice to ensure quality requirements. If the quality of requirements improves, the project failure rate will decrease.

# CHAPTER 2: LITERATURE STUDY

The purpose of this section is to survey all relevant information that has previously been published. The questions that the literature study attempted to address are as follows:

- What is a system in a software engineering context?
- What is requirements engineering in a software engineering context?
- How is requirements engineering done?
- How important is requirements engineering?
- What are the key challenges faced in requirements engineering that affect quality requirements?
- What are the key attributes of quality requirements?

The context information is surveyed as background. The scope of the literature study then focuses specifically on establishing the known factors that contribute to quality requirements during the requirements engineering process of a project and how quality requirements influence project success or failure.

## 2.1 Requirements Engineering

During the 1990s Gause and Weinberg (1990) stated that the *"fledgling problem solver invariably rushes in with solutions before taking time to define the problems being solved"*. In a software engineering context, requirements engineering is about defining the problem that must be solved, and then defining a solution (Cheng and Atlee, 2007). Jackson (1995) defines the requirement as the problem "in the world" and "the machine" as the proposed solution that is constructed. In this study a solution is an information system and its effective deployment into an operational environment to solve a key business problem.

The relationship between the problem world and machine solution is illustrated in Figure 3. It is described by Jackson (1995) and summarised by Van Lamsweerde (2009) as each having its own phenomena as well as sharing others. The shared phenomena are how the machine interacts with the world. The machine either controls or monitors the shared phenomena in order to solve the problem, i.e. implement the requirements. The requirements are concerned with the world phenomena, i.e. what the machine's effect is on the world as well as assumptions made about the world.

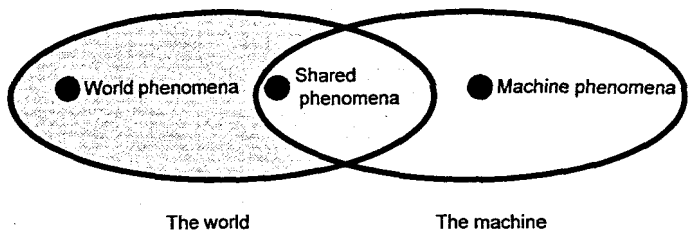


**Figure 3: Problem world and the machine solution (Jackson, 1995)**

In engineering the machine solution is referred to as a system. In the following section a system is defined and related to the engineering context.

## 2.2 Systems

The Institute of Electrical and Electronics Engineers (IEEE, 1990) defines a system as "*a collection of components organized to accomplish a specific function or set of functions*". Wasson (2006) describes a system as "*an integrated set of interoperable elements, each with explicitly specified and bounded capabilities, working synergistically to perform value-added processing to enable a user to satisfy mission-oriented operational needs in a prescribed operating environment with a specified outcome and probability of success*". As summarised by Van Lamsweerde (2009), a system can be seen as whole from the properties that emerge from the components' interactions.

To solve the world problem, an understanding is required of how the as-is system operates without a machine solution and how the to-be system should operate when the machine solution is implemented and operational (van Lamsweerde, 2009). To achieve this understanding, what the system does needs to be analysed, i.e. the system performs a function on the input and then produces an output. An analytical representation of a system is given by Wasson (2006) as illustrated in Figure 4. This analytical representation provides a checklist of all factors that should be considered when a system is specified.

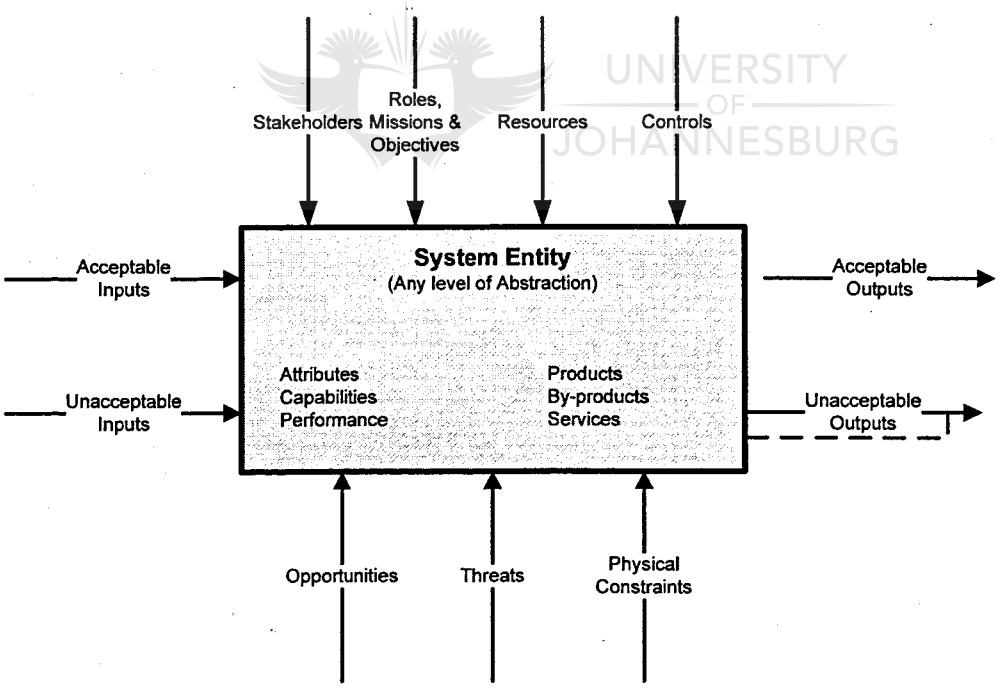


**Figure 4: Analytical systems representation (Wasson, 2006)**

There are different types of systems, for example economic systems, financial systems, cultural systems and government systems. Not all world problems would require a machine solution, but some systems would require engineering of systems (Wasson, 2006). A combination of engineering disciplines could be required, for

example software engineering, electrical engineering and mechanical engineering. This study focuses on a problem where the solution requires a system to be engineered.

## 2.3 Software Engineering

Software engineering is a discipline of software production from the initial stage when specifying the problem to be solved until the maintenance of the system when it is used by the users (Sommerville, 2001). The principal steps in a software engineering process are requirements, design, code, test and integrate (Sommerville, 2001):

- Requirements: Define the needs and constraints by analysing the requirements and interfacing with stakeholders.
- Design: Create possible solutions to meet requirements and select the best fit solution.
- Code: Software design is realised programmatically.
- Test: Testing is done to ensure that each component meets its specification.
- Integrate: All code or programs is/are integrated into a system to ensure that all requirements are met. A solution is delivered to the users.

Software engineering can follow different structured approaches; the next section describes some approaches currently in use.

## 2.4 Software Engineering Process Models

When implementing a solution, different process models can be used to implement a possible solution throughout the system's development life cycle. Software engineering is the process of creating or altering systems, and the models and methodologies that people use to develop these systems. The software engineering process model followed by an implementation team to implement a system depends on prior experience of the resources, standard approaches used by the organisation or problem type to be solved (Sommerville, 2001). A few process models are discussed in the following sections.

### 2.4.1 Waterfall Model

Each activity in the overall process is done as a separate sub-process. In this model as illustrated in Figure 5, the preceding step must be very close to completion before the next step starts. In practice when implementing complex systems, it is almost impossible to have requirements that do not change during the implementation life cycle (Dorfman and Thayer, 2000). The advantage of the model is that it is a simple process to manage (Sommerville, 2001).

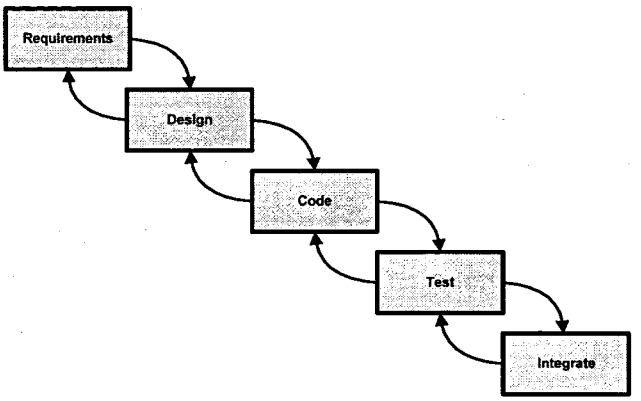


**Figure 5: Baseline management and waterfall models (Dorfman and Thayer, 2000)**

An iterative approach is used in the waterfall model where each successive step provides feedback to a previous step (Royce, 1970). However, the waterfall model is mostly interpreted as a sequential linear process (Boehm, 2006).

### 2.4.2 Prototyping

In high-risk implementations, especially where there is complex integration with high volumes of data between multiple systems, prototyping is used. Typically only one or two interfaces are constructed of the final system to simulate volumes and response times. Users can, after the prototyping, gain a better understanding of the final system and then provide better input into the requirements. Prototyping is also used in user interface applications by creating mock-ups and getting feedback from users before finalising all user interfaces. The prototype approach is illustrated in Figure 6.


**Figure 6: Prototyping life cycle model (Dorfman and Thayer, 2000)**

In prototyping an initial version of the software is produced; this is then used to demonstrate to users to gain a better understanding of the problem. It is a supportive tool to elicit input from users and validate requirements (Sommerville, 2001).

### 2.4.3 Incremental Development

Incremental development is very similar to what is also called evolutionary development. This approach is based on developing the system incrementally to satisfy the urgent immediate need, exposing it to the end-

users and refining it with many versions until the final system is completed. The incremental iterations are displayed in Figure 7.

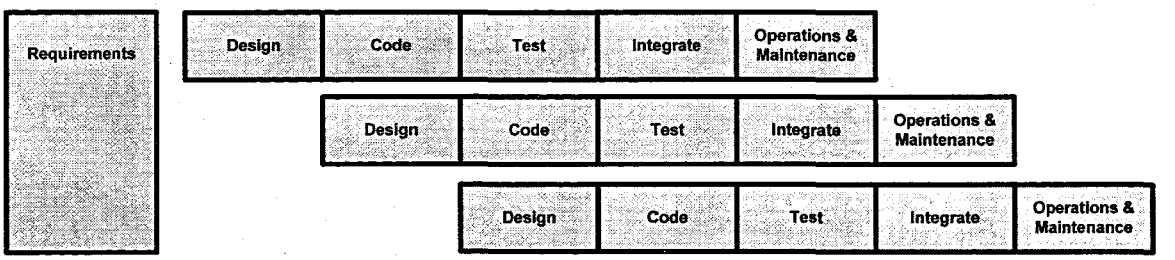


**Figure 7: Incremental development model (Dorfman and Thayer, 2000)**

The advantage of the incremental approach is that it allows for design changes or delays in finalisation of requirements. This could result in software that is structured inefficiently and difficult to maintain (Sommerville, 2001).

### 2.4.4 The Spiral Model

The spiral model addresses the problem of development cost estimation that cannot be done accurately. The approach followed in this model is a combination of all the other models but the main driver is risk (Boehm, 1986). The project is evaluated continuously after each phase. Each loop of the spiral in Figure 8 represents a phase in the project. After each iteration, changes can be made based on the re-evaluation. If the project needs to be terminated, this model will indicate it before all the phases have been completed.

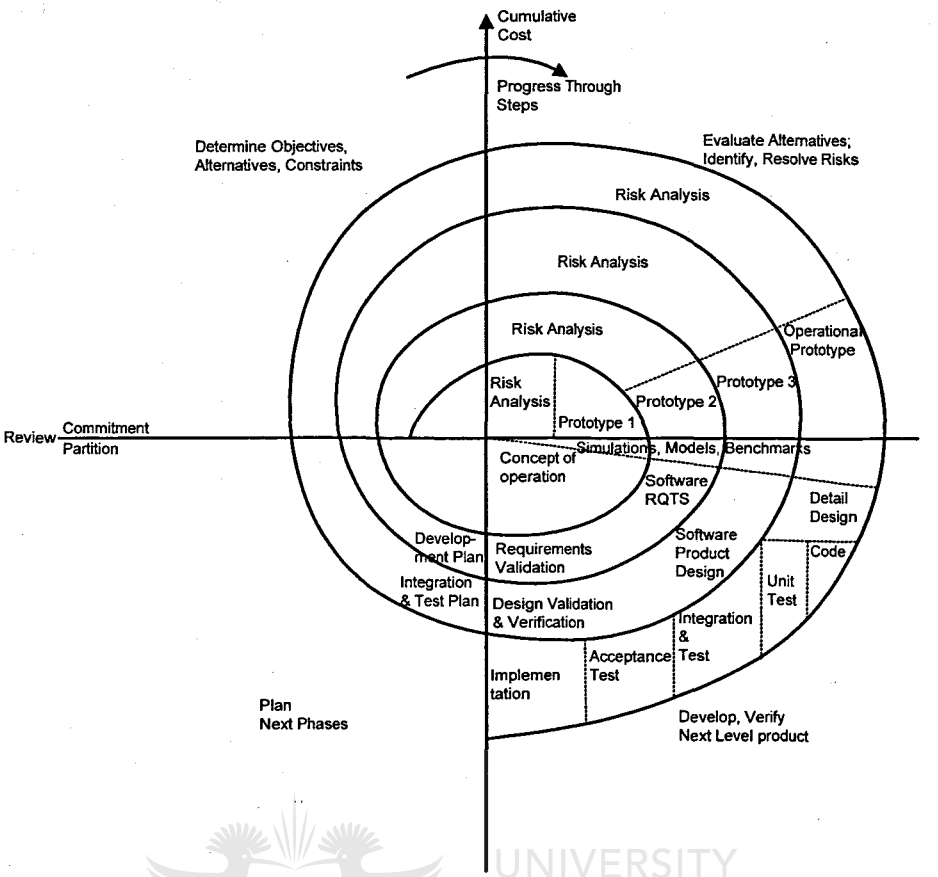


**Figure 8: Spiral model (Boehm, 1986)**



Some appropriate software engineering models that can be applied during system implementation have been discussed above with typical applications. The researcher concludes that during complex system implementation the spiral model enables customer satisfaction during delivery of a system as it continuously takes feedback and implements it. This is confirmed by a study of Van der Merwe (2002) that demonstrated that early prototyping and communication makes it possible to convert technology to demonstration product due to the circular nature of the process, as new knowledge can be implemented at a late stage.

With an understanding of where requirements fit into the software engineering context, the balance of this literature review focuses on defining requirements engineering as a subfield of software engineering and establishing what the known factors are that contribute to quality requirements during the requirements engineering process.

## 2.5 Requirements Defined

There must be an understanding of what the business problem is, i.e. what the machine's effect on the world is, before a solution can be designed. This includes the business goals that need to be achieved as well as the business drivers that drive the need for a solution. A thorough understanding of the operational environment is required to identify integration points (van Lamsweerde, 2009). Organisational standards are essential in the

identification of all the responsible and affected parties. The description of the business problem, why a solution is required and what the solution entails are documented in what is called a specification. A specification facilitates a common understanding between all the involved and responsible parties.

Figure 9 illustrates the requirements context. The original figure was used in a workshop by Regnell (2004), and Kamata et al. (2007) developed a requirements engineering metamodel based on this figure. This figure provides the context that should be considered when requirements are developed. It is about understanding the problem in the world, with all influences from the environment, and establishing a specification that must be validated by users. Feedback from validation is used during an iteration process to improve the specification. These requirements directly influence the design of the solution.

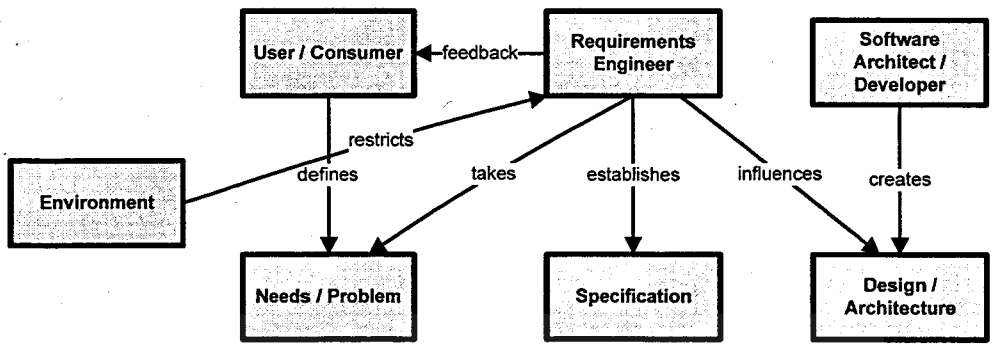


**Figure 9: Requirements context (Kamata et al., 2007)**

Requirements were defined in the 1970s by Ross and Schoman (1977) as *"a careful assessment of the needs that a system is to fulfil. [The assessment] must say why a system is needed, based on the current and foreseen conditions, which may be internal operations or an external market. It must say what system features will serve and satisfy this context. And it must say how the system is to be constructed"*.

Greenspan et al. (1994) define requirements as a specification of the system that must be developed. However, before an analyst can produce the requirements specification, an understanding is needed of the application domain in which the system will function, including the organisational environment. The specification needs to capture as much as possible of this understanding to support communication between all stakeholders, for example users, customers, developers and testers.

A more recent definition of a requirement is provided by the International Institute of Business Analysis (IIBA, 2009) as:

- *"A condition or capability needed by a stakeholder to solve a problem or achieve an objective.*
- *A condition or capability that must be met or possessed by a solution or solution component to satisfy a contract, standard, specification, or other formally imposed document.*
- *A documented representation of a condition or capability as in (1) or (2)."*

The IIBA based this definition on the IEEE Standard Glossary of Software Engineering Terminology (IEEE 1990).

## 2.6 Types of Requirements

There are generally two types of requirements:

1. Functional requirements describe the required to-be behaviour of the functions in the environment (van Lamsweerde, 2009).
2. A non-functional requirement describes constraints on how the functional requirements should be implemented (Pfleeger and Atlee, 2006; Sommerville and Sawyer, 1997). A classification of non-functional requirements is illustrated in Figure 10.

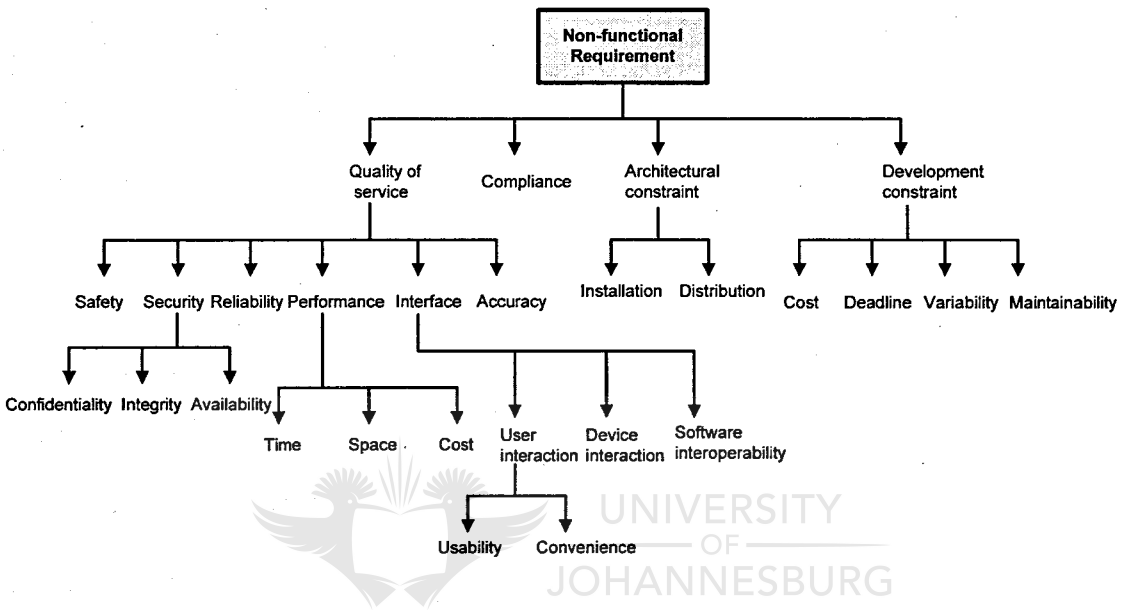




**Figure 10: Taxonomy of non-functional requirements (Van Lamsweerde, 2009)**

Functional requirements are typically the functions required by users. Non-functional requirements are more system related as illustrated in the above figure. In many cases, if non-functional requirements have not been considered, the system could be unusable, for example if the performance has not been considered, the functions will not operate correctly.

It is often assumed that stakeholders already know what the system requirements are at the beginning of a project. As far back as the 1970s Bell and Thayer (1976) cautioned that "*requirements for a system, in enough detail for its development, do not arise naturally. Instead, they need to be engineered and have continuing review and revision*". Requirements engineering and the process that facilitates the discovery of requirements is discussed next.

## 2.7 Requirements Engineering

Requirements engineering is not just a document, but a process of discovery. It is the process of activities that are performed to understand the problem that needs a solution. The requirements are then specified for a system that can solve the problem. When the specification is developed, different models or techniques could

be used, for example scenario walk-through, models or prototypes that must be validated by users. Feedback from validation is used during the iteration process to improve the specification. This iterative approach is illustrated in Figure 11.

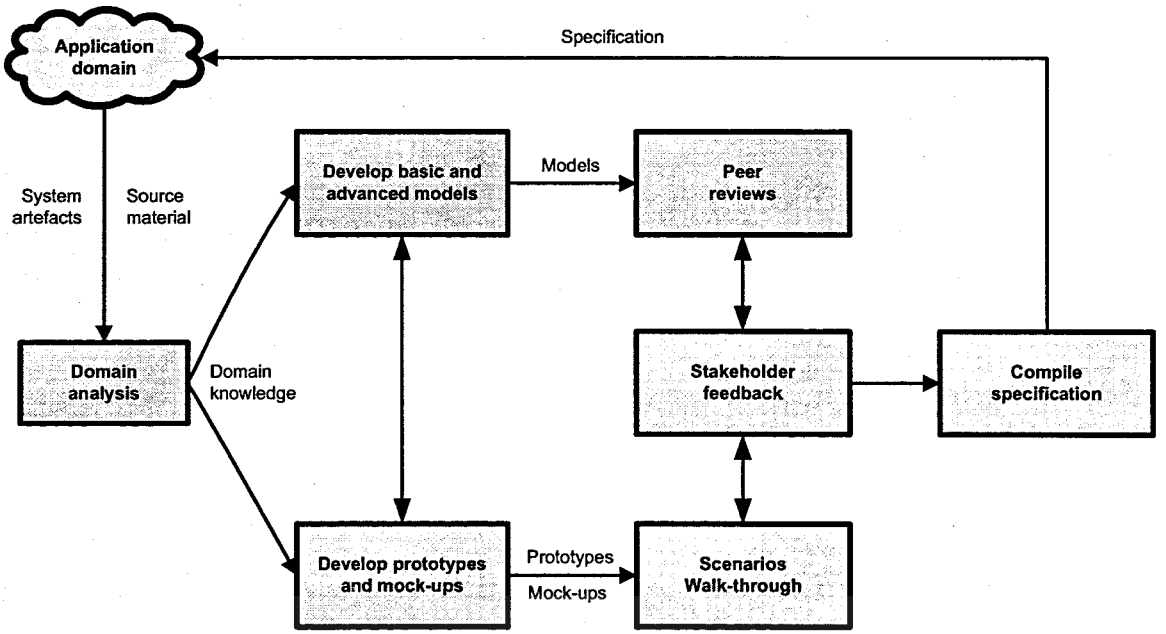


**Figure 11: Requirements engineering iterative approach (Hofmann and Lehner, 2001)**

The requirements engineering process is also described by Nuseibeh and Easterbrook (2000): "*The context in which Requirements Engineering takes place is usually a human activity system, and the problem owners are people. Engagement in a requirements engineering process presupposes that some new computer-based system could be useful, but such a system will change the activities that it supports. Therefore, requirements engineering needs to be sensitive to how people perceive and understand the world around them, how they interact, and how the sociology of the workplace affects their actions.*"

Requirement engineers use different tools and techniques to describe a system's proposed behaviour. Some of the techniques involved are (Hsia et al., 1993):

- Identification and documentation of user needs and problems. This will include a description of the "as-is" world of the user.
- Creation of the solution description, including the boundaries of the solution and what it will and will not do. This will include a description of the "to-be" world of the user.
- Analysis and validation of requirements documentation to ensure feasibility, validity, verifiability, completeness, consistency and traceability.
- Re-evolution and updating of changing end evolving needs.

In practice in many cases the requirements engineering discipline is not understood or seen as the capturing of a user requirement and moving the captured information to a development team. The previous sections show

that users do not typically know what their requirements are, and that it is a discovery process. The following section details what researchers have defined as focus areas during such a discovery process.

### 2.7.1 Requirements Engineering Process

Several taxonomies are used to describe the steps involved in the requirements engineering process. Sommerville (2001) describes them as system feasibility study, elicitation and analysis of requirements, specification of requirements and validation of requirements. Dorfman and Thayer (2000) define the steps as elicitation, analysis, specification, validation and management. The Software Engineering Body of Knowledge (SWEBOK) for software engineering describes the steps during requirements as fundamentals, process, elicitation, analysis, specification, validation and practical considerations (IEEE Computer Society, 2004).

All these taxonomies, if compared, are very similar. What should be covered in elicitation, analysis, specification and management is explored in the next section.

#### 2.7.1.1 Requirements Elicitation

Requirements elicitation is the discovery of requirements, where the requirements originate and how they can be collected. It is the first step in defining the problem in the world to be solved. The elicitation step is all about human activity, all the correct stakeholders should be identified, relationships must be established between the requirements engineer and the stakeholders. Knowledge will only be exchanged once these relationships have been established. The communication and common language established in these relationships will dictate the information exchange during elicitation (IEEE Computer Society, 2004).

(Kotonya and Sommerville, 1998) define requirements elicitation through four components:

1. Application domain knowledge is the part of problem world to which the system will be a solution.
2. Problem understanding is the specific customer problem; to gain this understanding the requirements engineer will build up knowledge on the domain.
3. Business understanding is required to understand how the system interacts and contributes to the business.
4. An understanding of the needs of each stakeholder (end-user, manager, business department, technical architecture etc.) must be obtained to know how the system should support the business operational processes and will impact the world.

The different components of in requirements elicitation are illustrated in Figure 12.

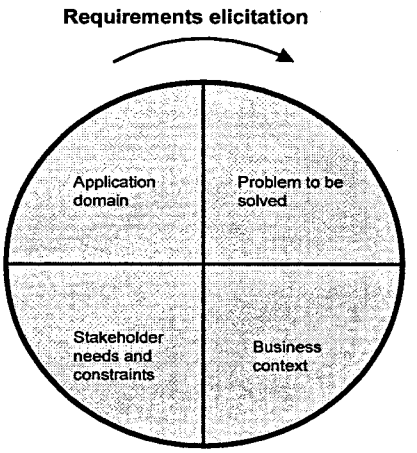


**Figure 12: Components of requirements elicitation (Kotonya and Sommerville, 1998)**

This multidimensional nature of requirements elicitation as illustrated emphasises that users do not know their requirements but it is a discovery process. To ensure complete requirements, all dimensions should be explored.

### 2.7.1.2 Requirements Analysis

As mentioned by the (IEEE Computer Society, 2004; van Lamsweerde, 2009; Sommerville, 2001), the activities involved during analysis are classification, negotiation including conflict resolution, prioritisation and requirements checking. The objective of the requirements analysis step is to identify problems in the draft requirements produced by the requirements elicitation step (Kotonya and Sommerville, 1998).

These requirements must be analysed to establish whether the implementation would be feasible, otherwise alternative solutions should be presented and the best options based on risks and goals should be agreed upon. The priority of each requirement should be discussed and agreed with the stakeholders to identify the most important requirements with the largest impact on improving the solution. Finally, the requirements must be validated for completeness and conflicts and should reflect what the stakeholders really desire.

### 2.7.1.3 Requirements Specification

This step documents the agreed requirements with supporting arguments of the system to be. This document contains objectives, definition, domain information, assumptions, constraints, functional and non-functional requirements. A wide range of techniques can be used during the documentation process, including natural language and modelling diagrams (Nuseibeh and Easterbrook, 2000).

### 2.7.1.4 Requirements Validation and Management

The final step is about quality assurance. The requirement document produced in the previous step should be reviewed and signed by all the relevant stakeholders (end-user, managers), including the technical team that will utilise this document as an input to the design of the solution. Prototyping could be used to demonstrate the

requirements to the users and this facilitates the beginning of change management as well as communication to ensure that all stakeholders have the same understanding of the solution to be provided to the problem.

The steps described above are all part of the requirements engineering process. These steps are done multiple times during which the quality of the requirements is improved. The typical results of such a validation process are a list of problems with agreed upon actions that must be resolved (Kotonya and Sommerville, 1998). This iterative process should continue until all the relevant stakeholders are in agreement that the requirements reflect their need correctly and accurately. Such an iterative process model has been developed by researchers and is discussed in the next section.

### 2.7.2 Requirements Engineering Process as a Spiral Process

It is very important to note that the steps in the requirements engineering process as discussed above are not sequential but an iteration of increments. Van Lamsweerde (2009) has adapted the spiral model of Boehm (1986) to illustrate the requirements engineering process as illustrated in Figure 13. It is necessary to point out that each new iteration can occur at a different stage of the software engineering life cycle. There is no difference in the steps involved during this iterative requirements engineering process and the steps discussed in the previous section. The multiple iterations are merely emphasised. During this requirement elicitation process the knowledge of the problem world in which the system will operate becomes clearer. Requirements must then be evaluated, classified and agreed upon during requirements analysis. Validation should be done to ensure quality requirements. If this process is repeated multiple times, a set of consolidated requirements will be generated.



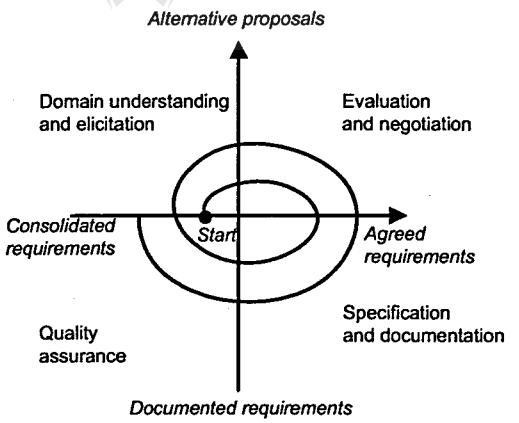


**Figure 13: Requirements engineering process (van Lamsweerde, 2009)**

From the above discussion it can be concluded that requirements engineering is not about communicating to users and capturing what the users think they need. It is a process to solve problems within situations of dynamic complexity. The following section describes how essential the requirements engineering process is during system implementation.

## 2.8 The Importance of Requirements Engineering

Requirements are the first step in the software engineering process and provide the basis for most activities during implementation. For example:

- Scope: Definition of the problem will be solved.
- Project management: All project costs, resources, delivery times are estimated based on requirements.
- Communication: Reference documentation to all parties involved.
- Technical: All software design/architecture, testing, documentation and training manuals are based on the requirements.
- Legal: If delivery partners are used, the requirements will be linked to a contract to state what needs to be delivered.

The reason why requirements engineering became so important is the changing nature of the industry and society in general (Berenbach et al., 2009). Product development life cycles have decreased; software is used as product differentiators and certain industries are regulated (Berenbach et al., 2009). If the requirements are wrong, an entire project is a failure even if all other work has been done perfectly (Viller et al., 1999). As mentioned by Brooks (1987), requirements are the most difficult to rectify and no other part in the software engineering impacts the resulting system as much as requirements do.

Errors in requirements cause systems to fail as the systems then do not function as intended or they do not match the users' or stakeholders' needs (Viller et al., 1999). In mission critical systems, for instance where safety is important, there is no margin for error in determining the requirements (Viller et al., 1999).

Quality has been determined as one of the main reasons why projects fail when the project deliverable does not function as required; the customer is then not satisfied and finds an alternative solution. It is concluded that errors in requirements contribute directly to quality of projects. In the next section the cost of errors in requirements is estimated based on other researchers' findings, after which the researcher will address the root causes identified of errors in requirements.

### 2.8.1 Requirements Engineering Errors are Expensive

Requirement errors are the most expensive errors in a project. If there are errors in the requirements, it has been estimated that the cost grows exponentially (Boehm, 1976; Boehm, 1984; Boehm and Papaccio, 1988). A later study by Westland (2002) supports this conclusion through the use of regression. Westland also demonstrated that errors become exponentially more costly with each phase of the software engineering process in which they were unresolved. The figures quoted by Boehm's studies are:

- 5 times more to detect and fix if they remain until design
- 10 times more if they remain until coding
- 20 times more if they remain until testing
- 200 times more after systems implementation

If quality requirements are delivered the first time the project will win on cost, time and quality. This statement by Hooks and Farry (2001) is supported by the figures given above. The more errors in the requirements that are discovered only in later stages of the system life cycle, the more cost will be incurred and the more time would be required for rework.

### 2.8.2 Role of Requirements Engineering in Cost Estimation

If there is no clear understanding of what needs to be solved, no concrete cost estimation can be provided either (Jones, 2007). Inaccurate estimations for system implementations have a negative impact on organisations. Overestimations could lead to a project not being approved for implementation and underestimation could get a project approved (Lederer and Prasad, 1992). However, when such a project is delivered, it will be over budget. In both cases, the organisation will not realise the anticipated benefits. Lederer and Prasad (1992) list the causes of poor cost estimations related to requirements as frequent requirement changes, user lack of understanding of own requirements, user communication and user understanding, poor specification of requirements and insufficient analysis.

Delivering quality requirements is important during a system implementation because it has a ripple effect on all downstream activities during the system development life cycle, as discussed in this section. The challenges to deliver quality requirements are identified in the next section.

## 2.9 Requirements Engineering Challenges



As noted in the previous section, there is evidence that requirements are one of the main contributors to system implementation quality. One of the first studies that explored the root cause of problems in requirements was a field study by Curtis et al. (1988), which found three of the most salient problems in software projects to be:

- The thin spread of application domain knowledge, i.e. understanding the problem in the world
- Fluctuating and conflicting requirements
- Communication and coordination breakdowns, i.e. how requirements have been communicated throughout the life cycle of the project

A study was done by Sutcliff et al. (1999) to identify problems in the requirements process during a specific project implementation where users did not want to use the delivered system. The problems identified were communication, social and organisational, with special mention of a lack of understanding and poor domain knowledge of developers, internal organisational politics and technical problems.

Hofmann and Lehner (2001) focused their investigation on how team knowledge, allocated resources and deployed requirements engineering processes contribute to project success. The main finding of the study was that successful project teams have in-depth knowledge about the application domain, information technology and the requirements engineering process. This is in line with the work of (Curtis et al., 1988). The teams that have implemented projects successfully have involved stakeholders throughout the life cycle of the project, maintained good relationships with these stakeholders and constantly validated their understanding of the application domain to avoid communication breakdowns.

A case study was used by Damian and Chisan (2006) to prove empirically that requirements engineering, if done effectively, can lead to benefits such as improved productivity, quality and risk management. Six principles, one of which was the use of cross-functional teams, were introduced to improve requirements and the benefits were observed. One of the main contributions to the benefits was collaboration instead of working in isolation where there was a united effort with no communication barriers. This supports the finding of (Sutcliff et al., 1999), which was that having rigid requirements engineering processes directly impacted on improvements in developer productivity, product quality and risk management.

Results from a survey by Karlsson (2007) on requirements engineering in a market-driven products environment indicated that the challenges faced are of an organisational and social nature rather than technical issues. These include issues concerning communication gaps between the business stakeholders and the implementation team.

A literature survey was done by Kamata et al. (2007) to describe what requirements engineering is, what it solves and what to focus on to improve the quality of requirements. The study concluded that formal methods for requirements elicitation have been developed by the research community, but they are not always practical. The study identified a gap in the research communities' work around communication during the requirements engineering process. Other findings are that research on non-functional requirements tends to focus only on security aspects and users' requirements must be understood through domain-specific knowledge on the business.

A literature review done by Walia and Carver (2009) classifies the requirement errors made during the requirements phase of the software life cycle to be people errors, process errors and documentation errors. The detailed classification is illustrated in Figure 14.

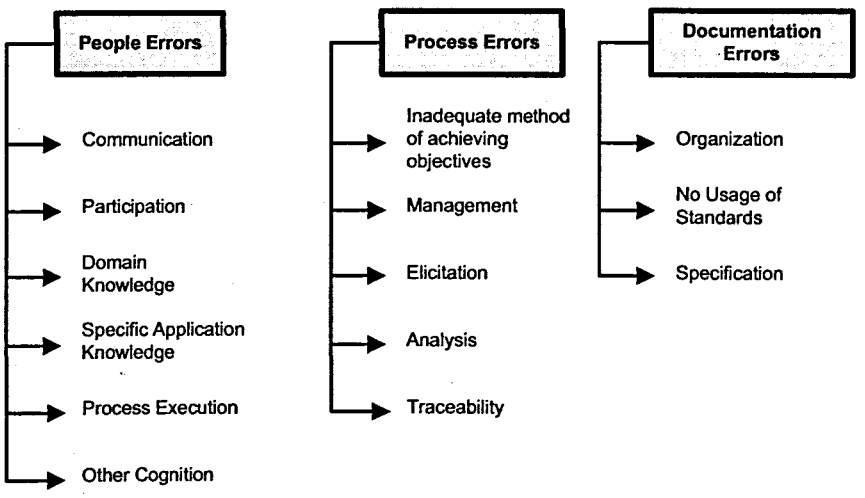


**Figure 14: Requirement error classification (Walia and Carver, 2009)**

All of these studies highlight a number of challenges that are experienced in practice in delivering accurate requirements. There are challenges that consistently appear in all the studies:

- Communication between the stakeholders and the implementation team that provides the solution
- Application of domain knowledge, i.e. understanding the problem in the world

If a communication breakdown occurs, no knowledge would be shared as the knowledge typically is with the domain users, i.e. typically the stakeholders. As previously mentioned, the context in which requirements engineering takes place is usually a human activity system. It is therefore a social activity, performed by individuals who carry out their work in an organisational context. These errors made by individuals during the requirements engineering process are classified into skill-based slips and lapses, rule-based and knowledge based mistakes (Viller et al., 1999). In the next section domain knowledge is defined, how it is acquired is investigated and the role it plays within requirements engineering is described.

## 2.10 Knowledge Defined

Knowledge is defined in the *Concise Oxford dictionary of current English* (Allen, 1990) as (i) expertise acquired by a person through experience; (ii) what is known in a particular field or in total; (iii) certain understanding as opposed to opinion.

Biggam (2001) has identified the following types of knowledge:

- Factual knowledge
- Practical knowledge
- Knowledge of people, places and things



Each type of knowledge can be derived from experience or rational thought, or from a combination of both. As long as the criteria hold that it must be true, the perceiver must believe it to be true and be in a position to know that it is the case.

Sveiby (1997) has formulated a working definition for knowledge "as the capacity to act effectively". Similarly, Davenport and Prusak (1998) argue that "knowledge can and should be evaluated by the decisions or actions to which it leads".

### 2.10.1 Characteristics of Knowledge

Sveiby (1997) determines that knowledge has four characteristics based on findings in (Polanyi, 1966).

- Knowledge is tacit – It is therefore very difficult to describe knowledge in words. We always know more than we can say.
- Knowledge is action oriented – We acquire new knowledge constantly through analysing sensory impressions. This dynamic quality of knowledge could be explained as follows: when an individual sees a set of data that reminds them of something familiar, the blanks are filled in. This integration of the mind is a skill a person has; each person needs to build this skill individually.

- Knowledge is supported by rules – Rules are consciously and unconsciously processing knowledge. Patterns are built up in the brain, and these rules enable us to act quickly and effectively without having to stop and think about what to do.
- Knowledge is always changing – When tacit knowledge is formulated through language it becomes static, and can then be distributed and increased.

From all the above facts on knowledge it can be concluded that a person needs to acquire the knowledge. A brief description of how knowledge is acquired is given in the next section.

### 2.10.2 Knowledge Acquisition

To acquire knowledge, we must integrate the new information into our existing knowledge and understanding. If there is no connection to our existing knowledge, we will simply remember facts but there will be no contribution to understand the world of that knowledge (Dawson, 2000).

As explained by Gharajedaghi (2006), to really understand a problem or concept it is not sufficient to elicit information. There are three levels required as illustrated in Figure 15:

- Information – Collect information on **what** the end-users do
- Knowledge – Build up knowledge on **how** the end-users do what they do
- Understanding – Understand **why** the end-users do what they do (all influence)

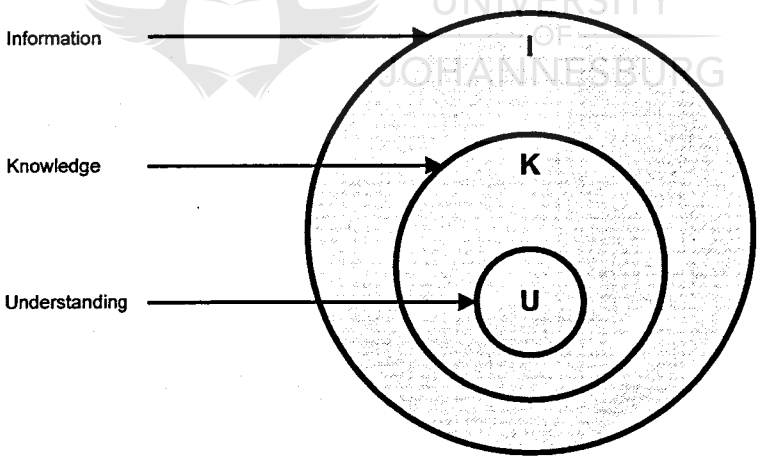


**Figure 15: Hierarchy of influence (Gharajedaghi, 2006)**

A definition of domain knowledge and why it is relevant to requirements engineering is given in the following section.

## 2.11 Domain Knowledge Defined

A domain is defined by Schreiber (2000) as a specific specialist field or discipline. Examples of domains are medicine, finance or chemical processes. Prieto-Díaz (1990) defines a domain within the context of software engineering as the application area for which the solution must be developed.

Kaiya and Saeki (2006) define domain knowledge as the knowledge of an expert in the domain. The knowledge of an expert is summarised (Alexander, 2003; Alexander and Murphy, 1998; Bransford et al., 2000) as follows:

- Experts possess extensive bodies of domain knowledge.
- They have the ability to recognise the underlying structure of domain problems.
- They have the ability to select and apply appropriate problem-solving techniques in the environment.
- They retrieve relevant domain knowledge with minimum cognitive effort.

Alexander (1992) defines domain knowledge as the knowledge individuals have about a particular field or subject. Her opinion is that it encompasses declarative knowledge (knowing that), procedural knowledge (knowing how) and conditional knowledge (knowing when and where). This knowledge operates on both the tacit and explicit level (Alexander, 1992). Ratchev et al. (2003) define domain knowledge within the requirements engineering context as *"a static knowledge and [it] consists of the concepts, relations and facts that are needed to reason about a certain application domain. It defines both the content and structure of domain knowledge in declarative form"*. As discussed in section 2.7.1.1 during requirements elicitation, domain knowledge is part of the world in which the system will be a solution. An understanding of this knowledge is required to understand how the system will interact with the business world.



Typically a requirements engineer will not have the domain knowledge and usually must acquire this knowledge from users in the domain (Zong-yong et al., 2007). Problem solving involves domain-specific knowledge and the use of domain knowledge in requirements engineering is explored in the next section.

## 2.12 Domain Knowledge Usage in Requirements Engineering

A range of knowledge is required during software engineering, i.e. software process and development languages. However, as described by De Oliveira et al. (2004), to generate quality requirements the following knowledge is also required:

- Knowledge about the domain in which the application/solution will operate
- Knowledge about the activities performed in this domain

In requirements engineering the domain of the problem is defined by Leffingwell and Widrig (2000) as the home (problems, culture, language) of the users and stakeholders whose needs/problems must be addressed to develop a system. It is therefore all about acquiring domain knowledge that forms part of the world in which the system will be a solution. During requirements analysis and elicitation it is important that the team acquire knowledge about the domain in which the solution will operate. Requirements elicitation cannot be solved with technology; it is all about how the knowledge is acquired in the social context (Goguen and Linde, 1993).

## 2.13 Conclusion

The chapter provided an overview of software engineering and described the relationship between software engineering and requirements engineering. The importance of requirements engineering was also highlighted as well as the consequences if this process is not executed properly.

The literature review provided valuable information on requirements engineering within the software engineering context. Two factors that contribute to the quality of requirements during the requirements engineering process constantly appear namely communication and domain knowledge. From this analysis of the literature, it was decided to investigate real-world projects that made use of requirements engineering as part of the project management process.

The investigation would provide insight into whether errors that were made during the requirements engineering process can be related to domain knowledge. This insight could provide guidelines regarding methods or techniques that can be used in practice to ensure quality requirements.

The following chapter focuses on the research method that needs to be used in order to address the problem at hand further.

# CHAPTER 3: RESEARCH METHOD

To investigate if errors made in requirements engineering during actual system implementation can be related to domain knowledge, an in-depth understanding is needed of the requirements process. A research method is required to assist in gaining this in-depth understanding. In the following section firstly alternative research methods are evaluated and a motivation is given for selecting a case study as the most appropriate research method to obtain answers to the research questions. Secondly, data collection and analysis during the research are explained.

## 3.1 Case Study Research

Case study research enables an in-depth understanding of a particular set of circumstances, in which multiple sources of evidence are used (Noor, 2008). It is a structured approach where the researcher collects all the data and interprets it (Simons, 2009). Case studies are used where the researcher is interested in gaining insight into and an understanding of why a specific instance happened as it did (Noor, 2008). It provides a rich description of an event (MacNealy, 1997). It is often suitable in software engineering research when the investigation is around how people work in teams to develop software (Host and Runeson, 2007).

## 3.2 Case Study Defined

A case study is formally defined by Stake (1995) as "the study of the particularity and complexity of a single case, coming to understand its activity within important circumstances". A second definition by Simons (2009) includes the purpose and research focus of a case study: "Case study is an in-depth exploration from multiple perspectives of the complexity and uniqueness of a particular project, policy, institution, programme or system in a 'real-life' context. It is research-based, inclusive of different methods and is evidence-led. The primary purpose is to generate in-depth understanding of a specific topic, programme, policy, institution or system to generate knowledge and/or inform policy development, professional practice and civil or community action."

Yin (2009) defines a case study as twofold: firstly, the scope of a case study is defined and then secondly, the technical characteristics are included as part of the definition:

1. *"A case study is an empirical inquiry that*
   - *investigates a contemporary phenomenon in-depth and within its real-life context, especially when*
   - *the boundaries between phenomenon and context are not clearly evident.*
2. *The case study inquiry*
   - *copes with the technically distinctive situation in which there will be many more variables of interest than data points, and as one result*
   - *relies on multiple sources of evidence, with data needing to converge in a triangulating fashion, and as another result*
   - *benefits from the prior development of theoretical propositions to guide data collection and analysis."*

The above quoted definitions all highlight one similarity, which is to understand a phenomenon or instance. Simply put by Gerring (2004), it is an "intensive study of a single unit for the purpose of understanding a larger class of (similar) units". In the following section the strengths and weaknesses of case study research are considered.

## 3.3 Case Study Research Strengths and Weaknesses

The strengths and weaknesses of case study research are summarised as follows from existing literature:

- A case study increases understanding of a particular issue (MacNealy, 1997). It is a detailed analysis of a particular situation, group of people or set of documents, which can lead to a more complete understanding of the aspect. The detail provides valuable insight into problem solving and evaluation (Cooper and Schindler, 2008).
- The detailed observation in a case study provides an opportunity to obtain a holistic view of a specific issue (Noor, 2008; MacNealy, 1997). It provides the context of the specific issue with all the detail of how and why things happen (Simons, 2009).
- A case study enables a researcher to obtain an in-depth understanding of a particular issue in the context of the organisation life (Simons, 2009). It provides insights into relationships within organisations which are difficult to access and are complex in structure (Easton, 2010).
- A strength mentioned by Flyvbjerg (2006) is the type of knowledge that is acquired from case studies. An expert's domain knowledge and experience are at the very heart of expert activity when using a case study as a research methodology. It provides knowledge about the practical (domain specific) which is more valuable than knowledge on the theoretical (general).



On the other hand, this methodology also exhibits some weaknesses that a researcher must take into consideration:

- One of the criticisms against case study research is that one cannot generalise on the basis of a single case and an option could be to use multiple cases (Yin, 2009).
- There is a perception that a case study could confirm the researcher's preconceived notions. (Flyvbjerg, 2006) has placed this in perspective: *"The case study contains no greater bias toward verification of the researcher's preconceived notions than other methods of inquiry. On the contrary, experience indicates that the case study contains a greater bias toward falsification of preconceived notions than toward verification."*
- Case studies are susceptible to poor research design (MacNealy, 1997). Researchers do not always follow systematic procedures or allow biased views to influence the findings and conclusions (Yin, 2009).

Practical and not theoretical knowledge is required to understand how quality requirements are obtained in practice. From the strengths it is evident that a case study could lead to an in-depth understanding of how to obtain quality requirements, provided that a systematic approach is followed to ensure good research design. Case study research is compared to other research methods in the following section.

## 3.4 Research Method Comparison

A summary is provided to evaluate when a research method is best suited for a situation by Yin (2009) as illustrated in Table 1. Three factors are used to determine which method is most appropriate for the particular research:

- The type of research questions which the researcher is investigating
- The control the researcher has to manipulate the actual behavioural events investigated
- The degree of focus on contemporary versus historical events

**Table 1: Relevant situations for different research methods (Yin, 2009)**

| METHOD | (1) Form of Research Question | (2) Requires Control of Behavioural Events? | (3) Focuses on Contemporary Events? |
|---|---|---|---|
| **Experiment** | how, why? | yes | yes |
| **Survey** | who, what, where, how many, how much? | no | yes |
| **Archival analysis** | who, what, where, how many, how much? | no | yes/no |
| **History** | how, why? | no | no |
| **Case study** | how, why? | no | yes |



The two research questions in this study were investigated to understand in-depth how quality requirements are achieved and why they are not achieved in particular instances in practice. Secondly, the purpose was to understand how quality requirements impact on project success. The research objective was to obtain multiple perspectives of a single process (the requirements process), over a period of time (system implementation life cycle). From the above summary, "how" and "why" research questions are more explanatory and favour case study, history or experiment methods. The reason for this is that the requirements process effects are not immediate and appear over time rather than at a certain frequency or incidence. History methods are used when dealing with the past, no persons to report what happened are available and investigations must rely on documents as main sources of evidence. The history method would not have achieved the objective of obtaining multiple perspectives of the requirements process and was therefore not considered. When using the experiment method, the researcher has control over actual events. During the requirements process a researcher does not have control and cannot manipulate the actual behaviour of the events investigated. It was therefore concluded that the case study research method would be best suited to achieve the research objectives as stated in chapter 1.

Poor research design and the fact that generalisations cannot be made on the basis of a single case were weaknesses mentioned in section 3.3. To ensure a solid research design and to ensure that these weaknesses did not influence the results of this research, a rigid systematic approach was followed. Two case studies were used to enable some generalisation. The systematic process followed by the researcher is described next.

## 3.5 Case Study Process

The important steps that should be followed in case study research have been taken from those mentioned by (Yin, 2009; Simons, 2009; Stake, 1995), and are as follows:

- Case study definition is where the research is planned and the research objectives are defined.
- Case study design is the selection of one or more cases and the definition of a study protocol to define how the data will be collected.
- Data collection is the collection of the actual evidence from the case(s) using multiple sources.
- Evaluation and analysis of the collected data.
- Reporting and deriving conclusions from the data.

The steps mentioned above are illustrated in Figure 16. Sections 3.6, 3.7 and 3.8 indicate how the researcher has structured the case study research using these steps as a basis.

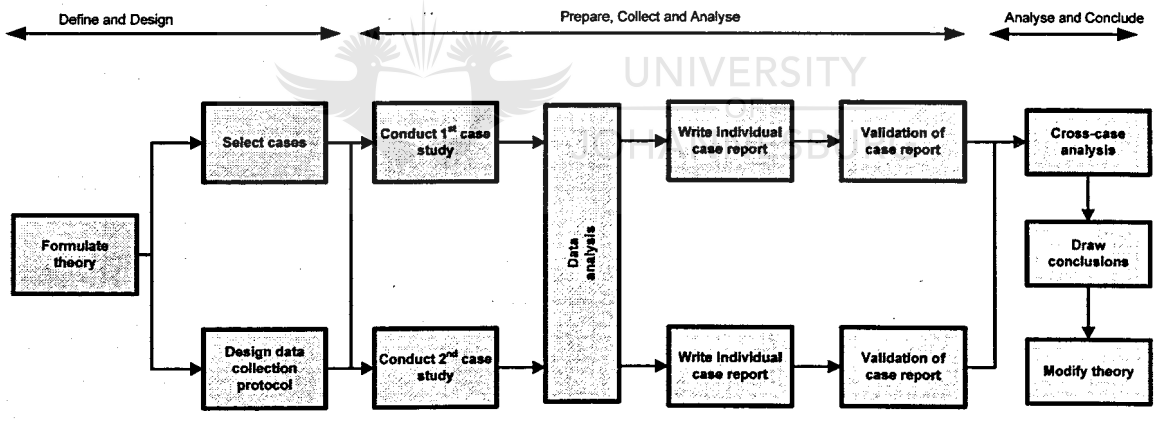


**Figure 16: Case study process (adapted from Noor (2008) and Yin (2009))**

How the processes were executed during the research is described in the following three sections.

## 3.6 Case Study Definition and Design

Stage 1 of the research includes formulating the theory, selecting the cases and designing the data collection. The formulation of the theory was described in chapter 2 which contained a literature study on the subset of requirements engineering. A common factor highlighted by the literature study, mentioned in section 2.11, that contributes to the delivery of quality requirements is the application of domain knowledge. The case study explores whether there is a correlation between domain knowledge and quality requirements during system

implementation in practice and whether this could be listed as a factor contributing to project success as mentioned in literature. Based on this the proposition of this research was to obtain data to support the assumption derived from the literature and additional real-world experiences of project failures and successes. The proposition states:

*If knowledge about the domain and activities in the domain is available, then quality requirements are derived.*

In addition to the study proposition, data was collected to answer the specific research questions as constructed in chapter 1:

- **RQ 1:** Why do some implementation teams produce good quality requirements and others poor quality requirements?
- **RQ 2:** How does the quality of requirements contribute to project success?

With the research questions and proposition defined, the basis of what data was relevant to the research could be derived. Selection of the cases to ensure that relevant data would be collected is described below.

### 3.6.1 Case Study Selection

The unit of analysis defines the "case" that will be studied. As the objective of this study was to obtain multiple perspectives of the requirements process during the system implementation life cycle, the unit of analysis is about the requirements engineering process. All cases selected therefore had to be about the requirements engineering process during system implementation.



This research explores whether there is a correlation between domain knowledge and quality requirements. The following data would therefore be relevant to provide insight and derive conclusions to prove this correlation:

- The investigation of factors that contributed to quality requirements during the requirements process of a system implementation
- The investigation of factors that contributed to poor requirements during the requirements process of a system implementation
- The investigation of factors that led to project success or failure

Based on the above factors during selection of cases, the following criteria were used to ensure that the cases had relevant data to collect:

- The requirements engineering process followed during system implementation had to have produced either good or poor quality requirements to enable the identification of factors contributing to quality requirements.
- The system implementation had to be completed to enable the investigation of factors that led to either project success or failure.

Two guidelines are provided by (Stake, 1995; Yin, 2009) when selecting case studies:

- Select cases where relevant data required is easily accessible (including the interviewing of people, review of documents and observation/participation in the field).
- Select cases where the maximum potential exists to derive data to answer research questions.

Based on the guidelines provided by these researchers, two individual projects were studied to understand a real-life system implementation. The two projects were selected within the same organisation and the same business unit. The researcher was a staff member in the organisation at the time and played an observation and participation role during both system implementation life cycles. Data and resources were easily accessible based on the fact that the researcher was a participant in the two projects. The selected support service unit had a vision to build a robust systems environment to meet both the operational and the strategic objectives. The business unit had no or limited systems in place, with none of the existing systems providing a system data view across all the business units within the organisation. The researcher agreed with the organisation that the organisation name will be kept anonymous.

Within Project XYZ a system solution was provided to multiple business units within the organisation. However, each business unit functioned as an autonomous business. The system was an off-the-shelf configurable system with a total budget of R100 million. This system was delivered into each business unit separately. The first business unit was very satisfied with the implemented system, but the second was not. Overall in the organisation the project was deemed very successful. The researcher felt that this case study would provide relevant data to understand the factors that led to project success.

Project ABC was an in-house developed system implementation with a total budget of R30 million. The system supported two business functions across the entire organisation. The requirements of one business function were said to be good quality requirements, but after delivering the system to the users this function had to be redeveloped as it did not support the business functions correctly. The requirements of the second business function were said to be poor quality requirements. Data was collected from this case study to understand the factors that contributed to the quality of the requirements. The overall system implementation was successful after some rework was done. Data was also collected to understand the factors contributing to project success that followed after rework.

With the cases identified as well as what data was relevant to find answers to the research question, the method and techniques used to gather the relevant data, including what data to collect, are described below.

### 3.6.2 Data Collection Protocol

This section deals with how data needed to be collected for the case study and which principles were adhered to.

**Principle 1 – Use of multiple data sources**

The researcher had to consider multiple sources of evidence for both cases. The sources that were selected are as follows:

- Documentation was selected as it can be reviewed repeatedly. The documentation also had a broad coverage regarding the project implementation and was available and easily accessible to the researcher. The following specific project documentation was collected: business case, high-level business requirements, functional requirements and project close-out reports. In addition, documentation on which

requirements process was followed, if any, was collected. The project documentation collected all consisted of signed-off versions.

- Participant observations were used as they provide reality, context and insightfulness into interpersonal behaviour and motives.
- To ensure that no manipulation by the researcher occurred, a questionnaire as a third data gathering tool was used to provide targeted information on the case study questions. The questionnaires that were used are in Appendix A.

**Principle 2 – Case study database**

A structured database was created by collecting evidence to assist the researcher during data analysis. However, this raw data collected was not available for independent inspection by other researchers. The reason for this decision is that the researcher agreed with all the participants that the collected data would be kept confidential. However, the findings from the collected data will be made available in a public report.

**Principle 3 – Chain of evidence**

A chain of evidence had to be maintained to ensure that the case study report contained sufficient citations to relevant data sources collected. For example, the case study database provided a reference as to whether the evidence retrieved was from specific documents, the questionnaire, observation or during participation.

Stage 2 of the case study research process deals with preparing to collect data, actual collection of the data, writing of individual case study reports and validating the case study reports. The following section describes the steps that are followed during stage 2 in more detail.



## 3.7 Prepare, Collect and Analyse

### 3.7.1 Data Collection Preparation

All participants were contacted prior to the research to obtain their cooperation. The business unit's chief operating officer responsible for delivering these cross-functional systems gave approval for this study in the organisation. The purpose of the study as well as required input were explained and all contact information was collected.

The following principles were agreed upon between the researcher, the organisation and participants (employees or ex-employees of the organisation):

- The organisation and participants' names would be kept anonymous.
- The data collected would be kept confidential but the findings from the collected data would be made available in a public report.
- Each participant was encouraged to be critical to ensure that the answers and opinions were objective.

- **Case 1 – Project ABC**

Fifteen project members agreed to participate in this study and fourteen responded. They included the business unit's chief operating officer, the chief information officer/programme manager, project managers, business

users, business analysts, the technical architect and developers. They were invited based on their knowledge of the life of the system implementation. This project had some employee turnover but participants were included that were part of every stage of the project.

- **Case 2 – Project XYZ**

Fifteen project members agreed to participate in this study and twelve responded. They included the business unit's chief operating officer, the chief information officer/programme manager, project managers, business users, business analysts, the technical architect and developers. They were invited based on their knowledge of the life of the system implementation. This project had minor employee turnover, but the majority of the participants included were part of every stage of the project.

### 3.7.2 Evidence Collection

The documents were sent electronically to the researcher. If additional documents were required, it was agreed with the business unit's chief operating officer that they could be requested from the project team members. Participant observation was based on actual involvement in the implementation of both projects as a staff member of the organisation. The questionnaire was emailed to all the participants with a requested return date. Once the questionnaires had been returned, the researcher coded and entered the data into a database so that it could be analysed independently or as an integrated whole. Integration took place once the case study progressed to the point of cross-case examination.



### 3.7.3 Data Analysis, Report Writing and Validation

The questionnaire responses were made anonymous and then loaded into a computer-assisted qualitative data analysis (CAQDAS) software package to analyse the questionnaires and documentation (Lewins and Silver, 2007). The CAQDAS package used was ATLAS.ti version 6. It enabled the researcher to code the questionnaires for analysis purposes. Coding facilitates the development of a detailed understanding of the phenomena which the data is seen to be presenting (Atherton and Elsmore, 2007). Participants completed questionnaires electronically, and no data interpretations were done by the researcher as the wording of participants would be used verbatim.

## 3.8 Analyse and Conclude

The final stage was analysing the case study evidence and drawing conclusions. The basis for the analysis was the theoretical proposition as explained in 3.6: *If knowledge about the domain and activities in the domain is available, then quality requirements are derived.*

(Yin, 2009) presents four general strategies that could be used during data analysis:

- Rely on theoretical propositions where evidence is then analysed based on the propositions
- Develop a case description which creates a framework for organising the case study
- Use qualitative and quantitative data including statistical analyses
- Define and test rival explanations

The data analysis strategy that the researcher used was based on a theoretical proposition as mentioned above. The researcher chose this strategy as it provides guidance for a data collection plan, it focuses the attention on certain data and ignores other useless data, and assists in organising the entire case study (Perry et al., 2005).

Five specific analytical techniques are available to assist during data analysis; they also enhance the development of internal and external validity (Yin, 2009):

- Pattern matching which compares an empirical pattern with a predicted one. Internal validity is strengthened if the patterns coincide.
- Data analysis by building an explanation about the case where patterns may be related to dependent or independent variables. The risk with this technique is the possibility of drifting away from the original focus.
- Time-series analysis tracking of trends over time. Intricate patterns can be followed, which leads to a firm foundation for conclusions.
- Logic models stipulate a chain of events over time in repeated case-effect-cause-effect patterns.
- Cross-case synthesis between the two or more cases, in which all the similarities and differences are categorised.

Pattern matching is used as a technique during data analysis, where a set of results will be predicted and then compared to actual results. As patterns begin to emerge, certain evidence may stand out as being in conflict with the patterns.



To enable conclusions to be drawn in order to state relationships in answer to the research questions, a systematic approach is required to make sense of data. Simons (2009) discusses some procedures that were utilised by the researcher in this study to ensure a systematic approach:

- Data reduction is the process of selecting data to focus on in questionnaires, observations and documentation. In this study, this was guided by questions posed in questionnaires. Once data has been collected, coding is done.
- Coding breaks the data into data segments and assigns a name to each segment. The data segments are then compared to each other and could be renamed if it would reflect more accurate naming.
- All coded data that relates to each other on a theoretical level is categorised.
- Issue and themes generation will happen as patterns begin to emerge during coding and categorisation of data.

## 3.9 Case Study Quality

To ensure that the case studies were of good quality four tests as given by Yin (2009) were applied. This ensured construct validity, internal validity, external validity and reliability of the study.

- Construct validity requires the researcher to use the correct measures for the concepts being studied.
- Internal validity demonstrates that certain conditions lead to other conditions and requires the use of multiple pieces of evidence from multiple sources to uncover convergent lines of inquiry.

- External validity reflects whether or not findings are generalisable beyond the immediate case or cases. Techniques such as cross-case examination and within-case examination, along with a literature review, help ensure external validity.
- Reliability refers to the accuracy of measurement. Case study design ensures that the procedures used are well documented and can be repeated with the same results over and over again.

Table 2 shows which approaches were followed in the case study to ensure validity and the section where the approach has been described is given.

**Table 2: Case study tactics for four design tests (Yin, 2009)**

| Test | Case Study Application | Section Addressed |
|---|---|---|
| Construct validity | • Multiple sources of evidence | 3.6.2 |
| | • Chain of evidence | 3.6.2 |
| | • Participants responses to questionnaires were completed electronically and no data interpretation was done; wording was used verbatim | 3.7.3 and 3.7.1 |
| Internal validity | • Pattern matching | 3.8 |
| External validity | • Literature reviews | 3.8 |
| | • Cross-case examination | 3.8 |
| Reliability | • Develop case study database | 3.6.2 |



## 3.10 Conclusion

The chapter focused on case studies as a research method and explained the process that a researcher must follow to ensure rigour and validity. It also indicated the strengths and weaknesses of this particular method.

Using a case study as the research method provides the researcher with an in-depth understanding of the problem at hand. In the case of this dissertation, case studies were used to determine why the requirements engineering process delivered either good or poor quality requirements. It is possible to gain knowledge of and insight into the practical world through the use of case studies. This knowledge would enable the researcher in future to develop effective methods or techniques usable in practice to ensure quality requirements. To ensure that this knowledge was usable, a rigid research process was followed to obtain trustworthy results.

# CHAPTER 4: CASE STUDY DESCRIPTION

In this section background on the two cases selected will be provided. A description is also given of how the case study evidence was collected and analysed during the rigid systematic approach that was followed, as discussed in chapter 3, to ensure a solid research design.

## 4.1 Research Objectives

The evidence discussed in this chapter is derived from two projects in the same organisation with the same support service unit. A five-year period from 2004 to 2009 was chosen for this study. The projects selected were both systems that had been implemented during the five-year period, when a requirements engineering process was followed during the systems implementation. During the requirements engineering process of each project life cycle the quality of the requirements was in some instances good but in others poor.

The goal of the research was to study the requirements engineering process through the entire project life cycle to identify the factors that contributed to quality requirements. One project delivered good quality requirements and the within the same organisation and the same business unit, with the same support structure, delivered poor quality requirements. Answers to address the following research questions were needed:

- *Why do some implementation teams produce good quality requirements and others poor quality requirements?*
- *How does the quality of requirements contribute to project success?*



The researcher's intention was to obtain data to support the assumption that *if knowledge about the domain and activities in the domain is available, then quality requirements are derived.*

## 4.2 Research Setting: The Support Service Unit and Its Requirements Engineering Processes

In this section background of the research environment is provided on the company and the support service unit where the case study research was conducted.

### 4.2.1 The Support Service Unit

The research was conducted in a support service unit of a financial organisation in South Africa. The financial organisation consisted of multiple business units across various sites with a support service unit providing guidance on the business processes and the governance over the business processes across all business units within the high-end market segment.

In addition to guidance and governance, the support service unit had to provide management with meaningful, consistent information about the values of customer operations across all the business units. The individual

business units in the organisation were responsible for implementing the business functions and processes supporting the functions in adherence of policies provided by the support service unit. This is illustrated in Figure 17.

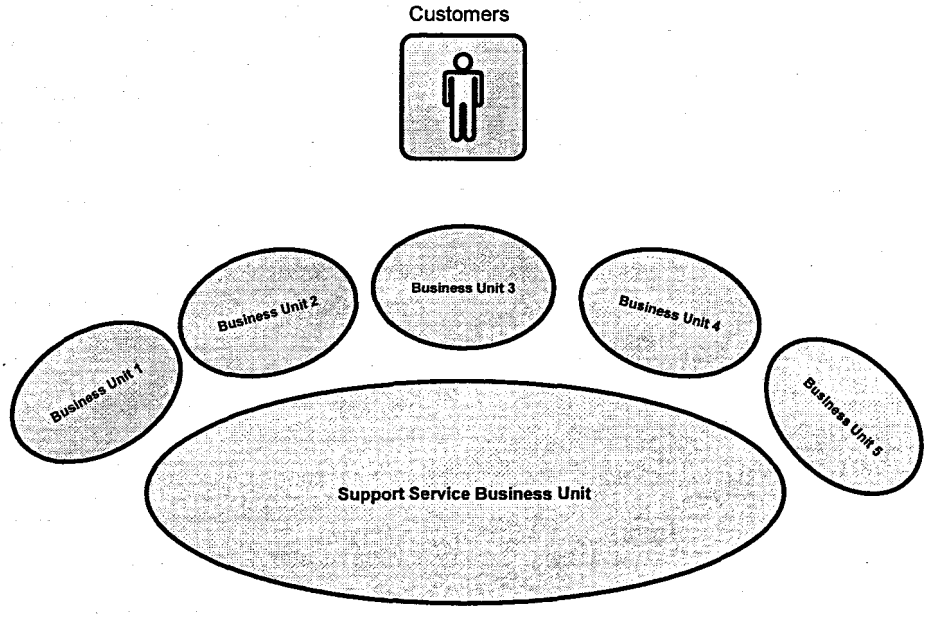


**Figure 17: Business operating model**

As illustrated by Figure 17, a single customer could deal with multiple business units as each business unit operates as an individual company with a different set of products and services. The biggest challenge was that there was no single consolidated view of a customer. In addition, the existing systems in the business units had various manual interventions which led to data integrity problems. The poor data quality issues were caused by the following:

- Each business unit identified the same customer using different identification approaches.
- There was no standardised approach to capture customers on data source systems.
- Source data was incomplete or incorrect and in some instances data was not captured at all.

Poor data quality led to unacceptable levels of operational risk, reputational risk, credit risk and operational efficiency. Poor data quality has a negative impact on the organisation's ability to make effective business decisions. The business functions impacted by the poor data quality issues were:

- Customer service – Poor customer data is one of the contributors to poor customer service, and this leads to the potential loss of business and market share.
- Regulatory reporting – The ability to comply with increasing regulatory pressures requires the organisation to be able to demonstrate that it has high standards in terms of coverage, completeness and correctness of data (Williams and Becker, March 2005).
- Operational efficiency – Poor data quality sources invariably lead to inefficient business processes, either through the duplication of effort or rework (Williams and Becker, March 2005).
- Strategic planning and decision making – Effective decision making requires that complete and accurate data be available to the executive management of the organisation (Friedman, 2002).

To address these data problems, the support service unit had to build a robust systems environment to enable the provision of meaningful, consistent information about the values of customer operations across all business units. The support service unit had no or limited systems in place, with none of the existing systems providing a consolidated data view across the various business units throughout the organisation. A few projects were initiated within a space of 18 months of each other to address 80% of the data problems.

### 4.2.2 Projects selected for Study

Two projects were selected to be studied as part of this research to determine why one project delivered good quality requirements initially and then poor quality after successful implementation in the first business unit and why another project within the same organisation and same business unit, with the same support structure, delivered poor quality requirements initially and then later on better quality requirements.

- Project ABC: This system provided input data to all the other systems in the environment. This project delivered a solution that collected, stored and managed the static data within the support service unit and business operations across the organisation. This project was initiated during 2005.
- Project XYZ: This was a second-order system which required data as input to perform calculations and provide standard output data to enable the downstream business processes. This project was initiated 2003.

The following two sections provide a summary background of the requirements engineering process during the system implementation life cycle of the project. The quality of the requirements produced during the execution of the requirements engineering process is discussed.



### 4.2.3 Background of Project XYZ Requirements Process

Customer acceptance of the system delivered was high after implementation in the initial business unit. However, customer acceptance of the system delivered in business units to followed was low. The requirements quality was very high during the implementation in the initial business unit but not complete during implementation in the business units that followed.

The points below summarise the requirements engineering context during implementation in the initial business unit. How the problem in the business world was understood, what influences from the environment were taken into account to establishing a specification and how users validated the specifications are covered.

- High-level requirements across all the business units were obtained by the project team from the support service unit. The approach that was followed by the project was the spiral method as depicted in section 2.4.4. This approach was selected to minimise the risk and to ensure that a complete set of requirements was available from all the stakeholders involved across the organisation. These included stakeholders from each business unit as well as sponsors and stakeholders from the support service unit. The team met with each business unit to elicit the business unit's specific requirements and gain an in-depth understanding of problems experienced in the business unit. Each iteration in the requirements process was triggered by the need to revise, adapt or extend the requirements by adding or modifying requirements and assumptions of domain-specific properties. After each iteration, changes were made based on the re-evaluation of

collected information. After nine business unit iterations, a consolidated view of the requirements was obtained and sessions were held with representatives from all the business units. The purpose of these sessions was to agree on priority of required functionality, reach consensus, resolve conflicts and validate the requirements.

- During these sessions with the various business units, the team formulated a set of definitions and business rules that facilitated communication between all the parties involved.
- To obtain the detailed requirements and calculation rules, group analysis sessions were held on a bi-weekly basis with all the users involved (stakeholders, managers, end-users, reporting users, legal users and all members of the project team). These users were only from the first business unit, but where functionality would impact other business units in future, users were included in this specific discussion. This was to ensure that standardisation across the organisation could be obtained. Each business function was decomposed and discussed.
- All the end-users in this business unit that formed part of these discussions had expert knowledge within the specific business unit of the products and services delivered. Some had knowledge at operational day-to-day level, some at daily management level and some at strategic decision-making level. Others were familiar with legal implications of decisions. The users had the ability to explain why they did what they did. They also knew what influenced the daily activities within the business unit. The users and the team had very good relationships established by now and information and knowledge were exchanged during formal sessions and informal conversation.
- The system was implemented as planned and the end-users were very satisfied with the delivery of the system. They trusted the system data from the day the system went operational. No rework was needed during the implementation. The users were happy with the system's functionality, which supported the business activities within the business unit sufficiently.
- Changes to the requirements requested after the implementation were as a result of functions that had been deprioritised by the business since the start of the project.

The points below summarise the requirements engineering context during implementation in the second business unit.

- The end-users in the second business unit did not need a solution as they had a self-created flexible system in place. This system was specifically developed to take care of the specific requirements of the individual business unit. Changes to the existing system did not require any change process and the developer of the system was part of the business unit. Changes to the system were made on a daily basis. The system did not follow rigid business rules as the developer had the ability to override these rules when required. A new system would remove this ability from the developer, as business rules would then be applied consistently across the organisation within the new system.
- After the implementation of the first business unit's system, the project team did not continue with group analysis sessions with all the end-users involved in the second business unit. The approach selected for this business unit was more one-on-one sessions with two main business users. One of the main business users understood the daily activities performed in this business unit in which the system would operate. He was involved in the high-level requirements session, was part of the system decision process and was included during the first business unit detailed analysis sessions to ensure standardisation across the organisation. This user moved into a new role in another business unit before the system for this unit was

implemented and operational. The second user was a more strategic user and left the organisation at the time the system went operational.

- Information was collected from these two users for detailed requirements but limited knowledge about the domain and activities in the domain was exchanged between the team responsible for the solution and the users within the second business unit. The main documents produced from these sessions were detailed requirements documents. The to-be business operation processes were only specified once the solution was implemented due to resource constraints within the team responsible for the solution.
- The system was implemented as planned. By the time the system was operational in the business unit, new resources had taken over the operational activities of the unit. The users did not yet understand how the system would impact the day-to-day activities in the business unit. There are various reasons for this: not all the processes had been specified by the team responsible for the solution; the users from the business unit that provided the requirements were no longer part of the business unit and the new users now had new requirements.
- The changes in requirements requested after implementation were due to knowledge not available about the detailed daily activities within the team responsible for the solution as the previous expert user from the business unit had moved into a new role. Changing and conflicting requirements were the order of the day and continued for two years after the business unit functionality was integrated into operations.

The requirements process during the first business unit was interactive with multiple users. A complete base of knowledge about the business domain activities was built by the implementation team. In contrast, the second business unit's requirements process followed a less interactive approach with individual users.



### 4.2.4 Background of Project ABC Requirements Process

Project ABC was perceived as a troublesome project that was turned around into a successful project after rework. This project delivered poor quality requirements and customer acceptance of the system delivered was low. After rework, quality requirements were produced and, once implemented, resulted in customers who were more satisfied.

The fundamental business driver for Project ABC was to enable the standardisation of business processes across the organisation where static data was collected and maintained for the purposes of the support service business unit, and the provision of structured data with the resultant improved management information system capability.

- A detailed as-is analysis was done to understand how each individual business unit implemented the business processes that had to be standardised across the organisation. These processes were documented with a good understanding of where the existing systems were used and what data was collected, and all problems experienced were identified. This detailed analysis was done across six of the main business units where the business processes for this business activity were executed and took about 12 months. At this stage very high-level requirements were presented to the business units to obtain consensus on them and approval to proceed to select a solution.

- At this point a decision was made to purchase a single technical solution for both Project ABC and another project responsible for managing the life cycle of supporting documents of the business process of Project ABC. These two projects were combined into a single project for the implementation phase.
- Once the business case was approved and implementation approaches agreed on, the detailed requirements and processes were defined.
- For Project ABC no structured approach to elicit the requirements was used. The requirements were based on what the current system did and merely fixing problems. One of the analysts had some knowledge about the business domain in which the application/solution would operate and the activities performed in this business domain. However, this knowledge was limited to the daily operational activities of one business unit.
- During the discovery of the detailed requirements the business analyst elicited information from a fellow business analyst who had some knowledge of the business activities and there were limited interactions with actual users of the system. There was no knowledge available at management and strategic level on the business domain and the impact of the system on other business processes within the support service business.
- Different terminology was used that contradicted the current set of definitions and business rules that facilitated communication between all the technical and business resources.
- The detailed requirements were circulated for sign off, at which point the shortcoming was identified that the business rules defined were the same as those for a previous obsolete system and contradicted new business rules.
- The project that was combined with Project ABC and that managed the life cycle of supporting documents of the business process of Project ABC used a more structured approach to elicit requirements. Detailed requirements were elicited during group analysis sessions with multiple users involved. The users were all from one business unit where the business process was executed. One detailed requirements document was produced from these sessions. No to-be business operation processes were specified. The requirements did not specify any data flow from input source.
- When implementation of these requirements started, multiple project resources left, including the project manager. With six months remaining to deliver, a project manager and business architect were reallocated to Project ABC to deliver the project with the remaining resources and some new analysts were recruited.
- A subject matter expert from business was allocated to Project ABC to assist the new team with queries about the requirements. The requirements for the initial Project ABC were reviewed and changes were raised to ensure integration into the business environment.
- By this time the development was complete for the part that managed the life cycle of supporting documents. The requirements did not include specifications on how screens should look and what data needed to be logically grouped together, so changes were requested on a daily basis. No design validation sessions were held between the developers and analysts, so all requirements errors were picked up during testing. This created an environment of constant changed requests when gaps were identified, crisis management, long hours, many defects and a lot of rework.
- It was also realised that the rework required on the functionality which supported the part that managed the life cycle of supporting documents was significant, that it would have to be specified again and redeveloped, that it did not support the business process at all and that it was not used.

- At this point it was decided to first solve the user interface problems and satisfy existing users before new business units were implemented. The requirements for new user interface were done in small focus groups with all users that used the system currently and in the future. The documents produced from the system were specified by incorporating input from strategic users that used the documents on a daily basis. Knowledge of how the activities were performed in this business domain was facilitated by the subject matter expert from business.
- To develop the user interface requirements a software engineering process model was implemented which incorporated a design validation session. Errors in requirements were picked up at the design stage, which created an environment with limited changed requests, fewer defects and limited rework if any.
- Once the changes on the user interface and documents produced went live, the users were extremely satisfied.

The requirements were initially elicited without user interaction. Once new resources had been assigned, including a subject matter expert, more interaction followed between users.

## 4.3 Research Data Collection

The multiple data collection methods included documentation inspection, participation, observation and the use of a questionnaire. The data procedures during the data collection process are explained below.

### 4.3.1 Documentation Inspection



Documentation produced during the requirements engineering process of both projects was studied. No formal requirements engineering tool was used in the environment and all requirements documentation was free-text formatted. However, a generic system development life cycle based on the SWEBOK of the IEEE Computer Society (2004) was adapted, with all mandatory artefacts required to be produced during the system development life cycle. The following artefacts were collected and studied:

- Vision document that facilitated agreement with business about which business objectives were to be supported by systems
- Business case documents to obtain financial approval after the system was selected
- Research paper detailing data quality issues faced in the business unit
- Position papers providing background of analysis and proposed implementation approaches
- High-level requirements used as input for system selection and for creating a business case
- Detailed requirements specifying functional requirements of all required functions
- As-is documentation describing the business process prior to system implementation
- Process documentation specifying business operation process impacts
- Change requests to document any change requested and implemented
- Post-project implementation review document from users after project implementation to document lessons learned

Secondary document sources such as presentations, release notes, emails and meeting minutes were also used. As rigid processes were followed to document all required artefacts during the requirements engineering process, historical data was available to track all decisions, communications and any relevant data.

### 4.3.2 Participation and Observation

The researcher was employed by the organisation as a team member of the support service unit during the five-year period from 2004 to 2009 of the study. During this period the researcher was actively involved in the requirements engineering process of both projects. The researcher's role and responsibilities on each project are summarised in Tables 3 and 4.

**Table 3: Project XYZ overview of researcher's role**

| Researcher's Role | Description | Project XYZ |
|---|---|---|
| **Participation** - Business analyst | Responsible for producing requirements, configuring system | March 2004 – August 2006 |
| **Participation** Project manager / Business analyst | Drive & manage the implementation | January 2007 – August 2007 |
| **Observation** Business architect | Review requirements and implementation impacts | August 2007 – February 2009 |



**Table 4: Project ABC overview of researcher's role**

| Researcher's Role | Description | Project ABC |
|---|---|---|
| **Observation** | Observe and provide assistance when required | March 2005 – August 2007 |
| **Observation** Business architect | Review requirements and implementation impacts | August 2007 – November 2007 |
| **Participation** Project manager / Business analyst | Drive & manage the implementation | November 2007 – February 2009 |

The researcher either played an observation or participation role during the projects.

### 4.3.3 Questionnaire

To ensure that no manipulation could occur due to the researcher's participation and observation role, the data gathering was followed up by a questionnaire. Refer to Appendix A for the questions. The participants were invited based on their knowledge of the requirements engineering process execution during the projects to ensure that there was representation during the entire system implementation life cycle. This was necessary because Project ABC had significant staff turnover.

Twelve out of the fifteen participants for Project XYZ completed the questionnaire and fourteen out of fifteen did so for Project ABC. The participants' role in the project and/or organisations are summarised in Table 5. Five participants completed both questionnaires due to their role in the support service unit, for example the support service unit chief information officer and the support service unit chief operating officer or some participants were reallocated from one project to another.

**Table 5: Overview of participants**

| | Project XYZ | Project ABC |
|---|---|---|
| **Participant Role** | **Number of Participants** | **Number of Participants** |
| Business analyst | 5 | 6 |
| Business expert (subject matter expert) | 1 | 1 |
| Developer | 1 | 2 |
| Project manager | 2 | 2 |
| Support service unit chief information officer | 1 | 1 |
| Support service unit chief operating officer | 1 | 1 |
| Technical lead | 1 | 1 |



The aspects of the requirements engineering process of a system implementation life cycle that were investigated using the questionnaire are described in the next section.

### 4.3.4 Questionnaire Aspects of Investigation

The questionnaire survey was conducted during 2010 to validate the researcher's findings during participation, documentation review and observation. As mentioned, the detailed questions are shown in Appendix A. The following subsections each contain a description of a category of questions in the questionnaire with the underlying rationale for asking the questions.

#### 4.3.4.1 General feedback on individual role during project

The proposition of this research was to obtain data to support the assumption: *"If knowledge about the domain and activities in the domain is available, then quality requirements are derived."* The researcher wanted to obtain general feedback from participants on how long they had been employed and what their role was on the project to ascertain if they had had time to gain an understanding of the business environment prior to project implementation. In addition, the aim was to obtain the participants' view of whether knowledge of the business activities which the system would impact had an influence on their individual performance or did not influence the quality of their work delivered.

#### 4.3.4.2 Requirements process followed during implementation phases

Wu et al. (2009) indicate that no one user in the organisation possesses all the knowledge required for a complete set of user requirements. Each user has a piece of knowledge required to be competent in his or her work. When all the knowledge is brought together a complete set of requirements can be derived. Hence the importance, as explained by Wu et al. (2009), of guiding and properly organising users to enable the implementation team to discover the full extent of the user requirements. Therefore if during the requirements process users are not organised properly and not all users are involved, a complete set of requirements is not possible. Indirectly depending on which users are involved or not during requirements discovery could lead to lack of domain knowledge that contributes to the quality of requirements.

The participants were questioned about their personal understanding of the requirements process and how this process was executed during the project life cycle, who was involved during the process and what each contribution was to the user requirements. Through this, the researcher attempted to show that having multiple users involved, each with their piece of knowledge, contributes to the quality of the requirements. In addition, if only one user is involved it leads to incomplete requirements.

#### 4.3.4.3 Quality of Requirements

To validate the researcher's view that parts of the requirements engineering process delivered good quality requirements and others poor quality requirements, the participants were asked if good or poor quality requirements were delivered during the system implementation life cycle. They were also asked to comment on what factors contributed to good quality requirements and what factors contributed to poor quality requirements. They were required to highlight the most significant factor that contributed to the requirements quality.

The participants were also questioned on the influence of the quality of the requirements on the overall delivery of the project. The researcher thus attempted to point out that poor quality of requirements generates rework, which has a ripple effect on the workload, cost and customer satisfaction.

#### 4.3.4.4 Customer satisfaction

The criteria for project success are whether the project is delivered on time and within budget and whether the system works as required. If the system does not work as required users typically are not satisfied. The participants were requested whether they perceived the project as successful and if the customers were satisfied with the system delivery. They were also asked to identify the main contributing factor to project success or failure.

The researcher attempted in this way to show the direct relationship between quality of requirements and customer satisfaction. If customers are not satisfied, they do not use the system and the project ultimately fails if it does not deliver any benefit to business.

## 4.4 Research Data Analysis

The completed questionnaires were loaded into a CAQDAS software package to analyse. This software provides a tool to researchers in order to manage their data analysis. It provides the capability to code and categorise large amounts of narrative text but it does not generate codes (Lewins and Silver, 2007). It enables the researcher to order and categorise the data faster than having to do everything manually (Atherton and Elsmore, 2007).

Qualitative coding is the process where data collected during the research process is classified in themes that relate to each other or relationships between issues are identified from the data. This enables the researcher to search for patterns in the data more easily (Lewins and Silver, 2007). Simply put, it allows the researcher to work more efficiently in an automated way to understand the *"'mess' and ambiguity of the data collected"* (Atherton and Elsmore, 2007).

The steps the researcher followed to code the data in CAQDAS were data reduction, data coding, categorising of data and lastly generation of issues and/or themes.

Data was generated to find evidence to study the proposition. The questions in the questionnaire dealt specifically with the requirements engineering process (unit of analysis) and the results of the requirements engineering process, as discussed in section 4.3.4, in order to reduce the amount of data generated.

A deductive coding approach was used, as the evidence that had been analysed was based on the study proposition. Computerised open text coding was done on each questionnaire by the researcher. The data segments that were considered for coding were identified from the research questions in section 3.6 as follows:

- In order to analyse data to find evidence to study the proposition
  - Impact of domain knowledge on requirements
- In order to analyse data to find evidence to answer the research question which states: Why do some implementation teams produce good quality requirements and others poor quality requirements?
  - Factors contributing to good quality requirements
  - Factors contributing to poor quality requirements
  - Factors contributing to project success
  - Factors contributing to customer satisfaction

Once all data had been coded 74 data segments were available. Data was then categorised into families of data, where segments of data that were related were placed together. The categories in which the coded data were categorised were as follows:

- Customer satisfaction; domain knowledge; employment information; factors contributing to poor requirements; factors contributing to quality requirements; impact of quality requirements; knowledge on business activities; project success; quality of requirements; rework.

To ensure the trustworthiness of the research, validation strategies were implemented. This also ensured quality of the data and the appropriate application for each strategy in this study. This is discussed in more detail in the following section.

## 4.5 Validity

A case study research method was used to increase understanding of a particular issue. To ensure that the results were quality and adequate and to ensure the study construct validity, internal validity, external validity and reliability, four tests were applied as identified by Yin (2009). Table 2 indicates which approaches were followed in the case study to ensure validity. The following four sections provide a detailed description of how these were applied.

### 4.5.1 Construct Validity

Construct validity refers to the operational measure that was established for the concepts studied. This is done to ensure that the researcher has accurately chosen what to measure correctly in line with what the study intends to measure. The measures taken and discussed next are based on guidelines provided by (Perry et al., 2005).

*Intentional validity:* Do the constructs we chose adequately represent what we intend to study?
The objective of this study was to identify the most significant contributing factors that differentiate quality requirements from poor requirements. The researcher used a literature review to guide the case study focus and, as discussed in section 3.6, elected to focus the research on gaining an in-depth understanding of why some implementation teams produce good quality requirements and others poor quality requirements.

*Representation validity: How well do the constructs translate into observable measures?*
Multiple sources of data were utilised as discussed in section 4.3 to ensure cross-checking of perceptions from different angles to strengthen the evidence. The participants completed all responses electronically and no interpretations could be made by the researcher as responses were used verbatim and not changed. The case study database contains all the references to keep track of the origin of the evidence.

### 4.5.2 Internal Validity

Pattern matching was used as a specific analytical technique which compared an empirical pattern with a predicted one. Internal validity is strengthened if the patterns coincide.

### 4.5.3 External Validity

The data analysis strategy that the researcher used was based on a theoretical proposition. A literature review was done to focus the case study research. In addition, cross-case synthesis was done between the two case studies and all the similarities and differences were categorised. This was done to enable generalisation and thus validate whether findings can be replicated across the different cases.

### 4.5.4 Reliability

To minimise errors during the case studies and ensure a solid research design, a rigid systematic approach was followed, as discussed in chapter 3. In addition, a structured database was created during evidence collection. This enabled a chain of evidence from an audit perspective if a finding needs to be tracked back to origination.

## 4.6 Conclusion

Two case studies were selected to investigate the requirements engineering process and to obtain an in-depth understanding of why the process delivered either good or poor quality requirements. The evidence was collected systematically and analysed according to a rigid systematic approach to ensure a solid research design as prescribed by the case study method.

Using a case study as the research method has increased the researcher's understanding of the reasons why the requirements engineering process delivered either good or poor quality requirements. It confirmed the researcher's preconceived notions; however, it identified underlying factors that were not considered or known by the researcher.

The acquired knowledge adds to the body of knowledge regarding requirements engineering. It highlights the importance of understanding the underlying factors that contribute to quality requirements. It has focused the researcher's thoughts with regard to future research to develop effective methods or techniques that can be used in practice to ensure quality requirements.



Chapter 5 contains the actual findings from each case study as well as a cross-case analysis to determine whether any generalisations can be made.

# CHAPTER 5: RESEARCH RESULTS

The data patterns identified during the analysis for each individual case study are discussed in this chapter. A cross-case study analysis will follow after the two cases in order to categorise all similarities and differences. To identify themes from the data the actual text was categorised into families based on the theoretical proposition. Once data had been categorised into families of relationships, themes were generated as patterns began to emerge. The following two sections contain the main themes of each individual case study separately.

## 5.1 Project XYZ Data Results

Five main themes were identified from the data retrieved from questionnaires completed by participants in Project XYZ. A summary of the themes are presented in Table 6.

**Table 6: Project XYZ themes**

| Themes | Description |
|---|---|
| Theme 1 | Domain knowledge impacted on requirements |
| Theme 2 | Factors contributing to quality requirements |
| Theme 3 | Factors contributing to project success |
| Theme 4 | Trusted knowledge-based relationships contribute to project success |
| Theme 5 | User satisfaction followed naturally as a result of interaction between users and implementation team |

Each theme is discussed in detail in the following sections.

### 5.1.1 Project XYZ Theme 1: Domain Knowledge impacted on Requirements

The literature highlighted a number of challenges faced in practice to deliver accurate requirements, as discussed in section 2.8. One key challenge that consistently appears is the implementation team's understanding of the problem in the world.

A total of 83% of the participants concurred with the literature that it is very important to acquire knowledge to gain an understanding of the business environment and thus improve their individual performance to deliver quality work. Figure 18 shows the distribution of percentages of participants. The two participants that indicated that this was somewhat important stated that they fulfilled a technical and managerial role and less detailed understanding was required from them.

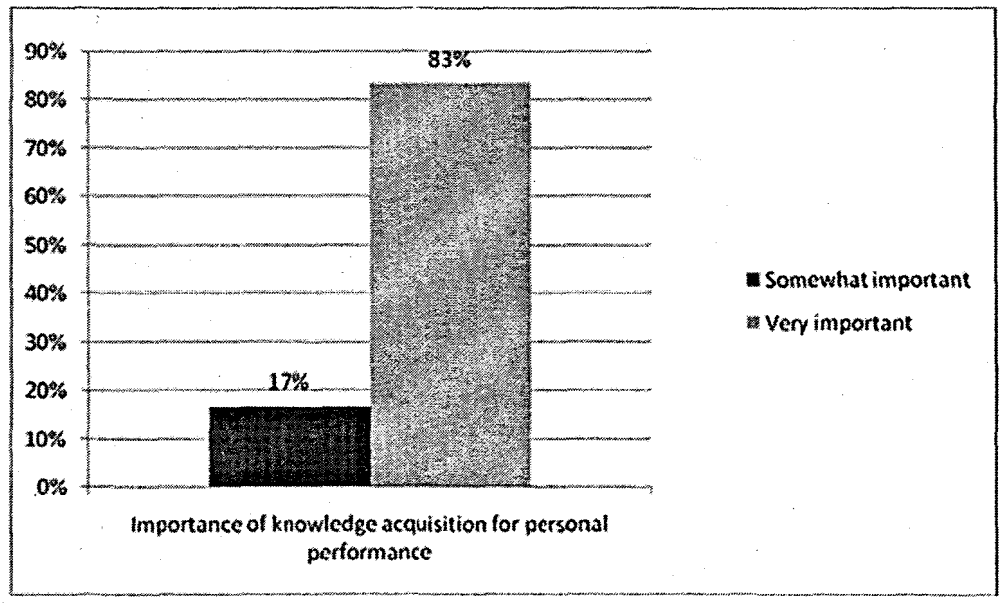


**Figure 18: Importance of knowledge acquisition for personal performance (Project XYZ)**

The project team members were in agreement with literature findings that domain knowledge impacts on requirements. Supportive quotations include: *"Without a thorough understanding of the processes and rules one cannot expect to fully understand the issues experienced by business and cannot add value or benefit. To create value an in depth understanding of the processes and rules is required."* More responses confirmed this notion:

- *"...analyst must understand the rules and processes to be able to design and implement a system that the users are able to use"* and *"If you have a poor knowledge of the process and rules you will not be able to fully understand the business and functional requirements."*
- *"It was vital to obtain this knowledge so that it could be translated into comprehensive and accurate specifications"*
- *"It is essential that one understands these processes and rules because that is the only way that one can optimise, improve and make recommendations to business going forward."*
- *"Providing appropriate technology to enhance productivity depends on the knowledge of what it is that employees do manually in order to offer a solution that will help them to perform their job better."*

The above comments all support De Oliveira et al. (2004) who state that to generate quality requirements the following knowledge is required:

- Knowledge about the domain in which the application/solution will operate
- Knowledge about the activities performed in this domain

This implies that if knowledge about the business activities is not available, the requirements derived will not be complete. Furthermore, if the team responsible for solution implementation does not acquire knowledge about the domain in which the application/solution will operate, they will not be in a position to derive quality requirements.

### 5.1.2 Project XYZ Theme 2: Factors Contributing to Quality Requirements

All participants were in agreement that implementation in the first business unit produced good quality requirements. The reasons provided for the good quality requirements were:

- *"All stakeholders and business users were involved in the process and gave valuable input"*
- *"... there was very good interaction with Business. Business bought into the project and made themselves readily available and interacted with the project team extensively."*
- *"users were involved and had buy-in: the problem was well understood."*
- *"Stakeholder buy-in."*
- *"Initial and ongoing negotiation and agreement of requirements but with clear target dates and limited ability to re-open issues."*
- *"Business engagement and very good business sponsors."*
- *"Business were able to verify and agree the requirements and this was delivered accordingly."*
- *"The quality of the analysis was influenced by the related subject matter experts that were assigned."*
- *"Business knowledge key to produce quality requirements detailed analysis and research and regular review by business users and project champion"*
- *"The project team went to great lengths to understand the project at all phases of implementation. This process was driven by key leaders in the team"*
- *"I think it is extremely important to ensure that there is stakeholder involvement. Without that, requirements are often misinterpreted and more often than not, the final outcome falls well short of the defined objectives and goals."*
- *"The project team was very close to the business and they constantly communicated their understanding of the requirements."*



From the above reasons it is clear that there was constant communication between business users and the implementation team to ensure a common understanding. This interaction was not just with a single user, but with all types of users, each of whom had a piece of knowledge different from the other. This enabled the team to derive a complete set of requirements. This data confirms the theory discussed in section 4.3.4.2 that if not all users are involved during the requirements engineering process, a complete set of requirements is not possible. Indirectly depending on which users are involved or not during requirements discovery, a lack of domain knowledge could contribute to the poor quality of the requirements.

The important implications that are deduced from the above are that to obtain quality requirements the following must be available:

- Knowledge about the business domain in which the system is to operate contributes directly to the quality of the requirements produced.
- The expert users in the domain must possess knowledge about the domain in which the application/solution will operate and therefore be able to share the knowledge.
- The knowledge acquired by the team must be facilitated and validated by two-way communication between business users and the implementation team.

After the first business unit's success, the system was implemented in additional business units. However, these business units produced poor requirements. Reasons provided by the participants for the poor quality requirements were:

- *"The project was not properly managed and users did not feel part of it ... the problem was not properly understood and what was implemented was not a 100% solution of what the business unit needed."*
- *"There was limited interaction with business to understand the business processes and rules. Also there was a key resource in business that left at the time of analysing the requirements. This impacted the quality of the requirements."*
- *"Difficult users caused the requirements to not be as clear as necessary ... not very good business input from the various business representatives."*
- *"If systems are developed without a clear understanding of the business process that underlies them, the system will fail; either because it will not work as expected or because the users will not want to use it."*
- *"the second business unit did not really buy into the system for operational management."*
- *"In the problem areas highlighted poor stakeholder engagement outside of the support service unit. Problematic stakeholders (attitude and competence)."*
- *"I don't feel there was enough interaction with Business. Business were not clear on their objectives. As a result requirements reflected as such. There was limited interaction with business to understand the business processes and rules. Also there was a key resource in business that left at the time of analysing the requirements. This impacted the quality of the requirements."*
- *"I feel there were big question marks over whether business in fact bought into the project for business unit two. Had there been more involvement from Business then the quality of requirements and objectives would have substantially improved."*



The factors contributing to both poor and quality requirements are as follows:

- If there is limited interaction between users and the implementation team, limited information is exchanged, and knowledge acquisition is limited. If knowledge is not acquired, there will be no understanding. This is supported by the hierarchy of influence (Gharajedaghi, 2006) as described in section 2.10.2.
- If domain knowledge is not available, quality requirements cannot be derived.

Based on the data analysis of both business units, it can be concluded that there are three important factors contributing towards quality requirements:

- Interaction between users, and not just a single user, but all users, each with a piece of the knowledge puzzle, is required during the requirements engineering process to facilitate information exchange and validation and thus ensure a single understanding to derive a complete set of requirements.
- Quality requirements are not possible if the implementation team does not gain an understanding of the business activities.
- Quality requirements are not possible if users involved in the requirements engineering process do not possess knowledge about the business domain.

### 5.1.3 XYZ Theme 3: Factors Contributing to Project Success

Although the second business unit requirement process faced challenges in user interaction, all 12 participants rated the project to be an overall success. This is shown through comments such as: *"the most successful IT project I was involved with"* and *"remains as a success and is still the benchmark for other projects".*

The factors identified by the participants as contributors to success had three main themes, namely the team, the quality of requirements and the interaction between business and the implementation team.

One of the three factors that contributed to the project success was the project team responsible for the implementation. Supportive quotes include:

- *"The project delivery team."*
- *"Team work and highly cross skilled team members."*
- *"Good quality of the project team"*
- *"The resources were skilled, committed and cared for the work (and the team). They always got the job done regardless of all the challenges."*
- *"The project team and the fact that the team stayed together throughout the project."*
- *"A truly dedicated project team that stayed reasonably stable for the duration of the project."*
- *"Teamwork"*
- *"Willingness of people to share information, communicate problems timeously and a team mentality."*
- *"Highly skilled and motivated project team"*
- *"Clear and defined roles and responsibilities."*
- *"Business analysts who understood operational processes"*



The second contributing factor mentioned was quality of requirements, with comments such as:

- *"Well defines requirements upfront"*
- *"Good quality requirements"*
- *"Good requirements that met objectives"*
- *"The success of the implemented project corresponds to the quality of user requirements as these are used for the configuration of the system and the unit testing of the solution."*
- *"the solution that was delivered was of good quality."*

The third factor mentioned by the participants was the interaction between the implementation team and business users, with supporting comments:

- *"Positive stakeholders (attitude and competence) make a successful project"*
- *"User engagement attitude and perception."*
- *"The relationship that existed between the project team and the business people."*
- *"Good Stakeholder interaction"*
- *"The combination of business knowledge, good leadership and determination helped the project to be a success."*
- *"Very good communication between Business and the project team"*

From the above, three contributing factors were derived as reasons why this project was successful:

- A project team that consisted of skilled resources, with clear roles and responsibilities defined, committed to getting the job done. It is important to note that this team and business users all worked towards a single goal during the implementation.
- Good quality requirements.
- Communication and interaction between the team and the business unit.

The implication is that for a project to be successful, interaction between users is required during the requirements engineering process to facilitate information exchange and validation to ensure a complete set of quality requirements. If the requirements engineering process is not properly organised and guided, i.e. the team does not have clear guidelines, roles and responsibilities, users will describe requirements incorrectly or inconsistently. This implications confirm those of Wu et al. (2009) as discussed in section 4.3.4.2.

### 5.1.4 Project XYZ Theme 4: Trusted Knowledge-based Relationships contributed to Project Success

According to Dawson (2000), worthwhile relationships are built on trust. Without trust neither party can add value to the other, and the relationship will not be beneficial to either. This is confirmed by Ayers (n.d.) who believes that lasting relationships are built on a foundation of trust. However, trust means different things to different people. Research has been conducted by Ayers (n.d.) to define the elements of trust as follows:

- *"Congruence: I say what I mean and mean what I say. I walk my talk. I am honest and ethical."*
- *"Openness: I am receptive to others' ideas and opinions. I am willing to disclose what's on my mind."*
- *"Acceptance: Who you are is fine with me. I do not judge other people."*
- *"Reliability: You can count on me to keep my commitments. I do my best at everything I do."*

The implementation team managed to establish trust relationships with business users during implementation in the first business unit. There was no "us" and "them", as they trusted each other with information. The relationship between business users and the team was identified by multiple participants as a main contributor to the project's success. Supportive quotations include:

- *"Clear communication from the start and well organised workshops establishes relationships and trust."*
- *"Establishing trust etc makes/breaks the team/project."*
- *"The relationship that existed between the project team and the business people."*
- *"Establishing good working relationships is vital."*
- *"The project team was seen to be part of business and deliver in business for business."*

Regarding implementation in the subsequent business units, relationships were not built on trust between business users and the team. Supportive quotations: *"The project was not properly managed and users did not feel part of it ... Entailed lack of trust and rework"* and *"Problematic stakeholders (attitude and competence"*. Although the project was delivered according to the requirements of management and was perceived as a success, the users in the relevant business unit did not perceive it to be as successful as the first implementation. Supportive quotation: *"...was less successful but was accepted and used in business although business still make use of various bespoke processes and databases to fulfil their daily tasks."*

The implication is that if a trust relationship is not established between users in the business domain and the implementation team, information is not exchanged. Interaction is limited and leads to less domain understanding, which ultimately impacts the quality of the requirements. Quality requirements are a factor contributing to the project's success, as discussed in the previous section.

### 5.1.5 Project XYZ Theme 5: User Satisfaction followed Naturally as a Result of Interaction between Users and Implementation Team

Participants felt that the users from the first business unit were very satisfied with implementation, but users from other business units were not satisfied. The users from the first business unit used the new system in their daily activities, but the other business units still did not trust the system and this caused many changes and rework. Supportive quotations:

- Business unit 1: *"very satisfied although the bedding down took some time"* and *"The customers were very satisfied as the system delivered what was agreed on."*
- Business unit 2: *"not satisfied, causing a 're-implementation' and making managing relationships very difficult."*

The reason why users were either satisfied or not was attributed to the level of interaction between the users and the project team during the implementation process. During implementation in the first business unit the following can be stated as derived from the data provided in section 5.1: (i) multiple users with expert knowledge contributed during the project, (ii) the team validated their understanding continuously with the users and (iii) there was daily communication between the users and the team.



When the system went live for the first business unit, the users had already accepted it as their own system.

- *"Users were very satisfied with the delivery of the system (system was nominated for an innovations prize). There was buy-in from business during this phase and large percentage of the users was involved during the whole project."*
- *"Users were happy to sign off implementation relatively soon after go-live."*

However, the total opposite was true for the second business unit. There was interaction with only one subject matter expert, who left the business unit before the system was operational, and little or no communication took place between the team and the users in this business unit. The users did not accept the system and therefore did not use the implemented system. Supportive quotations:

- *"Less buy-in from the users. Only one key subject matter expert was made available to the project that left the project before the implementation was completed. Few of the users were involved during the whole project. The head of the user's business unit also left shortly after the project implementation. Some of the functional requirements were questioned by the users that were not involved during the project implementation. This then resulted in some change request."*
- *"I do not think business completely bought into the system. This could have been due to lack of communication over agreements on processes to be followed with business. Another factor could be due to key subject matter expert leaving project during implementation."*

- *"initially there was resistance to the implemented system due to inadequate change management, change in key stakeholders"*

The implication is that if there is continuous interaction with users a mutual understanding develops throughout the requirements process of what is expected, and at the time of delivery the users are not surprised by what the solution offers, as they were part of the solution. Two important factors are highlighted here:

- Communication with all users throughout the requirements engineering process is necessary to develop mutual understanding.
- Communication ultimately facilitates the change that the new solution brings to the business domain.

The following section provides a summary of the data findings across the five themes to identify any patterns.

### 5.1.6 Project XYZ Data Findings Summary

Project success is dependent on quality requirements to be produced by the requirements engineering process. Requirements are identified in the literature as a main contributor to project failure as discussed in section 1.2. The literature highlighted a number of challenges that are experienced in practice to deliver accurate requirements (see section 2.9). The two challenges that consistently appear in all the studies are communication between the stakeholders and the implementation team that provides the solution and application of domain knowledge. The research results from Project XYZ confirm the findings from literature.



The first finding is that if knowledge of the business activities is not obtained by the implementation team, quality requirements are not derived. Domain knowledge can be identified as a contributing factor to quality requirements.

The second finding confirms the literature that if there is no communication between users and the implementation team, challenges are faced during the requirements engineering process. From Project XYZ data, where communication did not include multiple users, domain knowledge was obtained but some pieces of the information/knowledge were missing and incomplete requirements were derived that impacted on the quality of the requirements.

The third finding from Project XYZ is that once trust relationships are established between the business users and implementation team, formal and informal communication is established. This communication facilitates the knowledge exchange between the parties. If these relationships are not established, only occasional formal communication is established during which domain knowledge is obtained but some pieces of the information/knowledge are missing and incomplete requirements are derived and this impacts on the quality of the requirements.

The fourth finding from Project XYZ data is that if the users are involved during the requirements engineering process and constant communication between the implementation team and users takes place, the users are satisfied with delivery. The users gain an understanding during the process of what can be expected from the system and system functionality is within their expectations.

A main contributing factor that impacted the quality of the requirements in Project XYZ was domain knowledge. However, this was not the root cause of the quality of the requirements; communication and interaction between business users and the team responsible for requirements were the factors contributing to quality or poor requirements. Communication and relationships formed between business users and requirements engineers enable knowledge acquisition, leading to mutual understanding. Due to this mutual understanding, customer satisfaction is high when continuous communication and interaction have taken place during the requirements engineering process. If communication and interaction have not taken place, customer satisfaction is low.

## 5.2 Project ABC Data Results

Four main themes were identified from the data obtained from the questionnaires completed by participants of Project ABC. A summary of the themes is presented in Table 7.

**Table 7: Project ABC themes**

| Themes | Description |
|---|---|
| Theme 1 | Domain knowledge impacted on requirements |
| Theme 2 | Once project team and business unit worked towards a common goal, success followed |
| Theme 3 | Interaction between business unit and project team contributed to project success |
| Theme 4 | Users' satisfaction followed after usability rework |



The following sections provide a detailed discussion of each theme.

### 5.2.1 Project ABC Theme 1: Domain Knowledge impacted on Requirements

A total of 93% of the participants felt that it is important to acquire knowledge to gain an understanding of the business environment and thus improve their individual performance to deliver quality work. Figure 19 shows the distribution of percentages of participants. Only one participant indicated that this was not very important, as that participant played a managerial role and less detailed understanding was required.

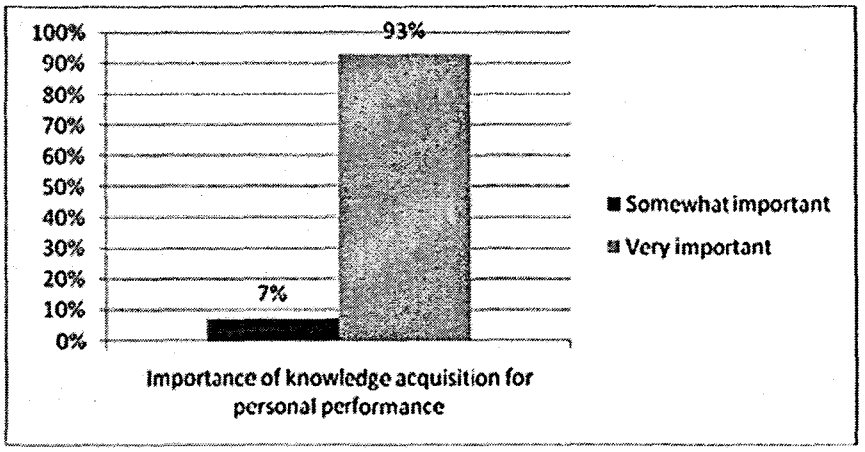


**Figure 19: Importance of knowledge acquisition for personal performance (Project ABC)**

Ten participants felt that the overall quality of requirements produced during the project life cycle was incomplete or poor but improved after rework. The main contributing factor to initial poor requirements mentioned was the lack of understanding about the domain in which the solution would operate. Supportive quotations:

- *"Individuals who were responsible for obtaining requirements did not have enough knowledge of the complete business processes"*
- *"the reliance on a single individual who had not worked in the business for at least 2 years but was held to be the expert had a negative impact on the requirements."*
- *"Initially project team BAs and PMs did not have a clear understanding of business problem - this was addressed with more focus and stronger guidance."*
- *"During the initial phase of the project the requirements were never formally documented as requirements and were also not signed off by the business stakeholders."*
- *"the lack of extensive engagement of different perspectives to the problem was in my view a factor that led to the poor requirement quality at the time that the build started."*
- *"...part of the project didn't appear to be understood properly and requirements weren't tightly defined leading to wide range of interpretations"*
- *"The biggest contribution in my opinion to the failure of the project was probably that there was a gap between what the users expected they were getting and what was documented"*

The participants felt that the quality of the requirements was poor but with rework with the addition of key resources that understood the business domain, this changed to good quality requirements. When questioned about what contributed to the quality of requirements, domain knowledge was identified multiple times as the single contributing factor. Some quotations:

- *"Without having the knowledge of the business processes and rules one cannot begin to understand where the issues that can be or need to be addressed are and how they can best be addressed."*
- *"Definitely having a dedicated business representative with an in-depth knowledge of the entire business process contributed to successful requirement gathering."*
- *"Key resources that understood the business processes as well as business did"*

- *"I think business knowledge is key hence having the correct people on the project is key."*
- *"Specifying the requirements correctly up front - although more work and time is spent in the beginning of the project acquiring the business knowledge, ensures that there are no changes identified further down the line when development is underway."*
- *"Validation sessions with the analysts have clarified a lot of the misunderstanding that may be introduced by language use."*

The implications of this data are summarised as follows:

- To derive a complete set of requirements multiple users should be involved during the requirements engineering process to ensure that the total range of knowledge is covered. This is supported by (Wu et al., 2009) as discussed in section 4.3.4.2.
- The impact of domain knowledge on the quality of requirements is emphasised once again.
- Knowledge acquisition about the business domain does not happen if there is no interaction between business users and the implementation team.

### 5.2.2 Project ABC Theme 2: Once project team and business unit worked towards a common goal, success followed

(Ayers, n.d.) believes that if there is no trust, there cannot be a relationship. As previously mentioned in section 5.1.4, if there is no trust relationship between two parties where each adds value to the other, the relationship will not be beneficial to either party.

From the participants' comments it is clear that initially the team and business unit did not work as a cohesive team with a single goal in mind. There were clearly no trust relationships between the business unit and the implementation team. Supportive quotes:

- *"I think this project was riddled with lots of hidden agendas throughout the project live cycle. This is not what one need on a project. The project team and business must work towards a common goal and should strive to achieve that goal together."*
- *"There were numerous scope and approach changes enforced on the project by the business, each time requiring a new approach to the requirements gathering."*
- *"Project manager was not able to guide the business analyst on what needed to be documented in the requirements specifications."*
- *"Self-importance in individuals: Individuals believing that they know more than they do"*
- *"At the time that the build started, the business refused to sign off the requirements which meant that in a very short space of time these requirements had to be reworked extensively as there were now severe time pressures on the project team."*

Once changes were made to the team and nearly the entire team was replaced, there was direction and collaboration between the team and the business unit, which enabled successes. The following supportive quotes are provided:

- *"Better direction and the joining of a business analyst with lots of business knowledge to indicate the gaps in requirements."*
- *"One focussed team pulling together"*

- *"The relationship that existed between the project team and the business people."*
- *"Commitment from both project resources and business"*
- *"Hard work, a team that stuck together in difficult circumstances, a business representative who was part of the project team and good leadership."*
- *"The ability of senior management to produce a single cohesive team, and co ordinate the work effort of many different skills and areas of focus was the main contributing factor."*
- *"The fact that business assigned a dedicated representative to the project and the fact that the project team understood the business well because of the long standing relationship that the project team had with the business."*

Effective knowledge communication and elicitation are based on communication (Dawson, 2000). This is a social process and if there is no trust relationship, this two-way communication will not take place. With no communication between business users and the implementation team, requirements cannot be complete. When trust relationships are established, collaboration follows naturally.

### 5.2.3 Project ABC Theme 3: Interaction between Business Unit and Project Team contributed to Project Success

When multiple project members left and many new resources joined the team with some resources reallocated from other projects to Project ABC, the project team interacted very closely with the business unit and very quickly established trust relationships. The team members were new to the environment and validated their understanding with the business unit. This interaction between the business unit and the team working towards a single goal was identified by participants as contributing to the project's success. The participants' supportive quotes:

- *"Communication between project team and business - regular and open meetings meant we all knew what was going on."*
- *"interaction between business and the project team to get a proper understanding of requirements, adaptability of project team to ever changing business environment."*
- *"The persistence of the project team and the ongoing engagement with business post implementation. Listening to business and implementing the things they wanted that will make the system work better and improve their day to day lives."*
- *"Solid commitment from key people in business to ensure that they system fulfilled genuine business requirements without losing focus and propagating poor business processes that may have existed with a previous system to the new system."*

Once regular interaction and communication between the business users and the implementation team were established, mutual understanding followed. This interaction was with the correct users who had the domain knowledge and this led to better quality requirements.

### 5.2.4 Project ABC Theme 4: Users' Satisfaction followed after Usability Rework

The participants indicated that the users were not immediately satisfied. There were usability issues with the system. The graphical user interface was not specified and the technical team developed it without any user input. After user groups were established and focused sessions were held to understand how the users used the system within their business process, the graphical user interface was reworked with the input from users and the system was accepted by them. Some supportive quotes:

- *"Somewhat satisfied. The users were very resistant to change and did not enjoy using the system, however did accept that it worked for them. After reworking the look and feel of the system the users were a lot more satisfied with it. This was mostly because the users themselves were used to identify look and feel requirements, so the issues they experienced with the system were eliminated."*
- *"The customers were not initially satisfied however the system improved over different delivery phases. Listening to their suggestions helped and making changes and meeting their requirements improved their satisfaction level."*
- *"Initially, no, because we had to roll out a system where the user interface was not optimally designed due to timing pressure to replace a legacy system. This was subsequently fixed and users became much more positive."*
- *"In the beginning we had a lot of resistance from the business users to use the system, but after a refactoring exercise and implementing two or three quick wins as suggested by business that system we accepted and is still in use today."*
- *"didn't hit the mark right off and had to tweak the system quite a lot in 'the real world' before users were happy."*
- *"Change is always a bigger challenge than expected and people react negatively to new systems without really giving it a chance."*



The requirements engineering process is a social interactive process: if users are not involved and do not validate the requirements, this will probably lead to a system that is not what users expected or want to use. A system can only be seen as successful if the users use it and do not find an alternative. As explained by (Hofmann and Lehner, 2001), successful teams repeatedly validate and verify requirements with multiple users, which leads the team to understand how the system will satisfy user requirements.

The following section provides a summary of the data findings across the four themes to identify any patterns.

### 5.2.5 Project ABC Data Findings Summary

The research results from Project ABC confirmed the literature findings that communication and domain knowledge play a very important role in achieving quality requirements.

The first finding is that if multiple users are not involved during the requirements engineering process, the problem is not understood, the implementation team will not have a comprehensive understanding of the business domain and poor quality requirements will be delivered. A lack of communication contributes to lack of domain knowledge, which in turn contributes to poor quality requirements.

The second finding confirms that effective communication is dependent on trust relationships, and collaboration follows only once this is in place. Once trust relationships were established between the business user and the implementation team, knowledge started to be exchanged. The fact whether a trust relationship exists between the implementation team and the business users can be derived as a contributing factor to quality requirements.

The third finding from Project ABC is that if interaction between business users and the implementation team does not take place, domain knowledge will not be obtained and requirements will be incomplete or poor quality. This communication facilitates the knowledge exchange between the parties.

The fourth finding from Project ABC data is that if users are not involved during the requirements engineering process, they will not be satisfied with the system delivered.

The data from Project ABC shows again that quality requirements are dependent on domain knowledge acquisition. The data confirms the fact that a comprehensive understanding of domain knowledge is gained by interacting with multiple users who possess the knowledge as discussed by (Wu et al., 2009). It emphasises that knowledge acquisition is based on two-way communication. This is a social process and if there is no trust relationship, this two-way communication will not take place. With no communication between business users and the implementation team, requirements cannot be complete. When trust relationships are established, collaboration follows. Once regular interaction and communication between the business users and the implementation team were established, mutual understanding followed, which led to customer satisfaction.



## 5.3 Cross-case Analysis

Data from Project ABC and Project XYZ is compared in this section to identify any cross-case patterns or differences. Table 8 provides a summary of the data from each separate case study from which the cross-case patterns were deduced.

**Table 8: Cross-case summary**

| | Project XYZ | Project ABC |
|---|---|---|
| Good quality requirements | • A well-organised requirements engineering process was established.<br>• Good relationships existed between multiple users and the team.<br>• Daily interaction and communication took place during the requirements engineering process.<br>• During implementation at the first business unit, the implementation team managed to acquire domain knowledge. | • A well-organised requirements engineering process was established only after the original requirements were delivered. Once the requirements process was established, it was used to facilitate changes to requirements.<br>• Good relationships between multiple users and the team were established once rework started.<br>• Daily interaction and communication took place during the requirements engineering |

| | | |
|---|---|---|
| | | process once rework started.<br>• Domain knowledge was acquired once rework started. |
| Poor requirements | • A well-organised requirements engineering process was established.<br>• A good relationship was built with a single user.<br>• There was limited interaction and communication.<br>• Domain knowledge was acquired. | • There was no organised requirements engineering process.<br>• There was a very limited relationship with users.<br>• Interaction and communication were limited.<br>• It was assumed that a single user/team member had the domain knowledge. |
| Customer satisfaction | • Users were involved during the requirements engineering process and understood exactly what would be delivered and how it would impact their daily business activities. | • Multiple users were involved during the rework of requirements. Customers were satisfied with the system only after changes were implemented. |
| Customer dissatisfaction | • Single users were involved and left before implementation was completed.<br>• Changes followed and have not stopped.<br>• Users found alternatives.<br>• There was no trust in the team or system. | • Users were not involved during initial requirements gathering.<br>• Changes were required.<br>• Users had no alternative but blamed the system for any non-performance on their part. |



The first pattern identified is that quality requirements are a result of interaction between users and the implementation team, facilitated by a well-organised requirements engineering process. Both projects delivered quality requirements at some stage of the project life cycle. The key contributor was the fact that once a well-organised requirements engineering process and good interaction between multiple users were established, the quality of the requirements improved.

The second pattern identified is that domain knowledge is a contributing factor towards delivering quality requirements. However, this is dependent on the interaction and communication between the users and the implementation team. If interaction or communication is limited, the implementation team does not have comprehensive domain knowledge and poor requirements follow.

Pattern three is that poor requirements are a result of limited interaction and communication between either just one user or the incorrect users, which leads to incomplete knowledge of the domain. The differences between the two case studies were that Project XYZ had a well-organised requirements engineering process whereas Project ABC did not, there was limited interaction only with individual users and both delivered poor quality requirements at a stage independent of a well-organised requirements engineering process.

The fourth pattern is that if users are involved during the requirements engineering process, constant interaction and communication between the users and the implementation team are established. The result is mutual understanding, and customer satisfaction follows automatically.

The final pattern is that customer satisfaction is very dependent on whether the users are involved during the requirements engineering process. If constant communication takes place between the implementation team and users, the users know what to expect. If users are not involved during the requirements engineering process, they will not be satisfied with the system delivered.

## 5.4 Conclusion

The patterns identified during the analysis of each individual case study were presented as well as a cross-case study analysis with findings. The case study proposition was confirmed: "*If knowledge about the domain and activities in the domain is available, then quality requirements are derived.*" However, the findings from both case studies showed that communication and trust relationships between the users and the implementation team are the main factors that contribute to the quality of requirements. If domain knowledge is not available, it is due to communication that has not been established.

Chapter 6 concludes the research and focuses on the contribution of the research.

# CHAPTER 6: CONCLUSIONS

In this chapter, the researcher reviews whether the study achieved its objectives. The findings of the study are concluded and recommendations discussed. In the last section areas for future research are suggested.

## 6.1 Introduction

The study main objectives were to identify the factors that contribute to quality requirements during the requirements engineering process of a project and to understand the impact of quality requirements on project success or failure. The study further explored whether there is a correlation between domain knowledge and quality requirements. The study results did indeed identify the factors contributing to delivery of quality requirements and how these affect project success. A relationship between domain knowledge and the quality of requirements was confirmed, indicating that if there is a lack of domain knowledge, quality of requirements is not achieved. These findings are discussed in the next section.

## 6.2 Findings

The research results show that domain knowledge is a recurring problem during the requirements engineering process and it affects the quality of requirements, thus impacting project success. Although domain knowledge is a main contributing factor towards quality requirements, it is not the root cause of the quality. Rather, this is a consequence of communication and interaction not having been established effectively. This implies that communication is the root cause of poor quality requirements. These consequences are illustrated in Figure 20.

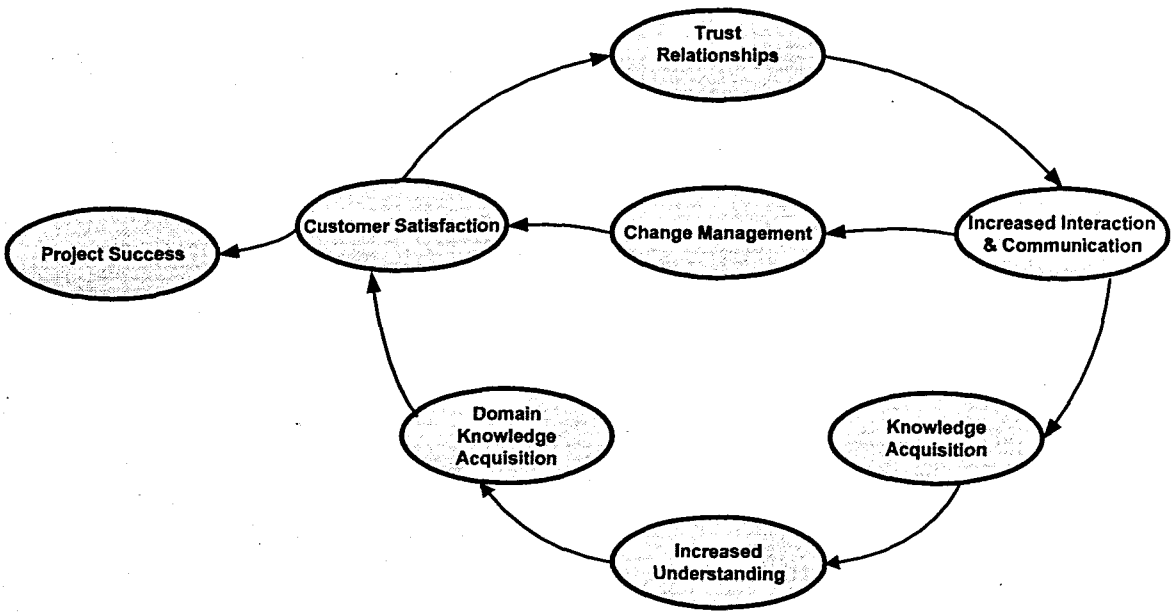


**Figure 20: Communication and trust relationship circle of influence**

The establishment of a trust relationship between users and the implementation team is essential. Without trust there is no communication and without communication there is no knowledge transfer. This leads to the team not understanding the problem, which leads to a lack of domain knowledge. Domain knowledge contributes to quality requirements, implying that if there is a lack of domain knowledge, the quality of requirements will be poor. Without trust, project success becomes a matter of luck (Hoffmann and Lescher, 2009).

Requirements emerge from the social interaction and communication between the users and the requirements engineer (Siddiqi, 1996). To master the requirements engineering process, a good requirements engineer needs to manage the incompleteness of communication (Rupp, 2002). Regular and open communication requires interaction with users on a continuous basis. This means staying in close contact with the users. If this does not happen, knowledge will not be exchanged and will result in either an incomplete or incorrect set of derived knowledge, which impacts the quality of requirements.

If there is no regular communication with users, they do not know what to expect from the delivered system and they will not understand why it affects their daily business activities. Once the system is delivered, they will not accept it and customer satisfaction will be a major problem. Open and regular communication facilitates the change management process.

Domain knowledge contributes directly to the quality of requirements, but to acquire domain knowledge there needs to be a trust relationship between the users and the implementation team. Once this is established, regular interaction and communication must take place which will facilitate understanding of the problem and the environment, and acquiring the knowledge of business activities so that the implementation team can identify all potential influences. For the user, this communication will facilitate an understanding of how the system will impact the daily business activities; this will also provide the users with the ability to confirm that requirements are reflected correctly.

It is the responsibility of the requirements engineer to manage and initiate this communication. If he/she does not have the ability to communicate or does not know what to communicate, the requirements engineering process is at risk of delivering poor quality requirements and the project is at risk of being delivered unsuccessfully.

Some possible approaches are now recommended that could be investigated to be developed as methods to facilitate communication.

## 6.3 Recommendations and Limitations

Excellent communication is an ability that can be learned by anyone. Methods and tools exist for communicating more effectively with different types of people (Cerri, 2000). A requirements engineer who does not know what to communicate becomes a problem as he/she is responsible for managing and initiating this communication. The focus of the recommendations is to suggest approaches on how to know *what* to communicate and not *how* to communicate.

Due to the dynamic nature of the requirements engineering process, systems thinking is suggested as an approach to know what to communicate during the formulation of the requirements. Systems thinking has been developed as an alternative method to analysis when solving a more complex problem. Analysis per se is the process of breaking the problem down into parts or sections and explaining the behaviour of each part as separate entities. The final step of analysis is then to attempt to aggregate the understanding of the separate parts into an explanation of the entire problem. Analytical methods are designed to handle complex problems with many variables (detail complexity). It is not designed to solve problems within situations of dynamic complexity that are based on cause and effect, are subtle or not obvious (Senge, 2006).

Systems thinking, on the other hand, explains the system in the context of the bigger picture and studies the role it plays within this larger environment (Gharajedaghi, 2006). Systems thinking is also defined as the discipline of seeing wholes. It is a framework for seeing all the interrelationships, for seeing patterns of change rather than static "snapshots" (Senge, 2006).

Systems thinking strives to understand how one entity influences another within the larger environment. It is therefore a framework that is based on the belief that each element of a system can best be understood in the context of its relationships with each other and with other systems, rather than in isolation.

If a requirements engineer uses systems thinking to understand the context of the business activities and to generate questions that should be asked, improved communication can be initiated.



## 6.4 Future Research

Systems thinking explains a system in context of the bigger picture and studies the role it plays within this larger environment (Gharajedaghi, 2006). Systems thinking is also defined as the discipline of seeing wholes. It is a framework for seeing all the inter-relationships, for seeing patterns of change rather than static "snapshots" (Senge, 2006).

From the definitions above, systems thinking strives to understand how one entity influences another entity within the larger environment. Systems thinking is therefore a framework that is based on the belief that each element of a system can best be understood in the context of its relationships with each other and with other systems, rather than in isolation.

Detailed research is suggested to investigate how to apply systems thinking and generate a framework for requirements engineers that can improve their ability to initiate the communication.

A second research direction could be how interpersonal skills could assist requirements engineers to manage the communication process. Engineers naturally possess technical and conceptual skills and generally lack interpersonal skills (Van Der Molen et al., 2007).

The successes of an organisation, team or project all depend on the knowledge in the organisation and are represented by the relationships between the people and the organisation (Dawson, 2000). Research could be focused on how to build these trust relationships.

Finally, South African tertiary institutions focus on software engineering as a discipline within formal curricula with limited coverage of and exposure to requirements engineering. This phenomenon could be investigated with a view to including requirements engineering as an individual module to prepare prospective requirements engineers for practice.

# REFERENCES


AGARWAL, N. & RATHOD, U. 2006. Defining 'success' for software projects: An exploratory revelation. *International Journal of Project Management,* 24, 358-370.

ALEXANDER, P. A. 1992. Domain Knowledge: Evolving Themes and Emerging Concerns. *Educational Psychologist,* 27, 33.

ALEXANDER, P. A. 2003. Can We Get There From Here? *Educational Researcher,* 32, 3-4.

ALEXANDER, P. A. & MURPHY, P. K. 1998. The research base for APA's learner-centered psychological principles. *In:* LAMBERT, N. M. & MCCOMBS, B. L. (eds.) *How students learn: Reforming schools through learner-centered education.* Washington, DC: American Psychological Association.

ALLEN, R. E. 1990. The Concise Oxford Dictionary. *The Concise Oxford Dictionary of Current English.* Eighth ed.: Oxford University Press.

ATHERTON, A. & ELSMORE, P. 2007. Structuring qualitative enquiry in management and organization research: A dialogue on the merits of using software for qualitative data analysis. *Qualitative Research in Organizations and Management: An International Journal,* 2, 62-77.

AYERS, K. E. n.d. Building Relationships in Selling. Available: www.integroleadership.com.

BELL, T. E. & THAYER, T. A. 1976. Software requirements: Are they really a problem? *Proceedings of the 2nd international conference on Software engineering.* San Francisco, California, United States: IEEE Computer Society Press.

BERENBACH, B., PAULISH, D. J., KAZMEIER, J. & RUDORFER, A. 2009. *Software & Systems Requirements Engineering: In Practice,* New York, McGraw Hill.

BIGGAM, J. 2001. Defining knowledge: an epistemological foundation for knowledge management. *In:* Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001. 7 pp.

BOEHM, B. 2006. A view of 20th and 21st century software engineering. *In:* 28th International Conference on Software Engineering Proceedings, 20-28 May 2006, 2006 New York, NY, USA. ACM, 12-29.

BOEHM, B. W. 1976. Software Engineering. *Computers, IEEE Transactions on,* C-25, 1226-1241.

BOEHM, B. W. 1984. Verifying and Validating Software Requirements and Design Specifications. *Software, IEEE,* 1, 75-88.

BOEHM, B. W. 1986. A spiral model of software development and enhancement. *In:* International Workshop on the Software Process and Software Environments, 27-29 March 1985, 1986 USA. 22-42.

BOEHM, B. W. & PAPACCIO, P. N. 1988. Understanding and controlling software costs. *IEEE Transactions on Software Engineering,* 14, 1462-77.

BRANSFORD, J. D., BROWN, A. L. & COCKING, R. R. 2000. *How People Learn Brain, Mind, Experience, and School,* Washington, D.C., National Academy Press

BROOKS, F. P., JR. 1987. No silver bullet; essence and accidents of software engineering. *Computer,* 20, 10-19.

CERRI, S. 2000. Effective communication skills for engineers. *In:* Engineering Management Society, 2000. Proceedings of the 2000 IEEE, 2000. 625-629.

CHENG, B. H. C. & ATLEE, J. M. 2007. Research Directions in Requirements Engineering. *2007 Future of Software Engineering.* IEEE Computer Society.

COOPER, D. R. & SCHINDLER, P. S. 2008. *Business Research Methods,* Boston, Mc Graw Hill.

CURTIS, B., KRASNER, H. & ISCOE, N. 1988. A field study of the software design process for large systems. *Commun. ACM,* 31, 1268-1287.

DAMIAN, D. & CHISAN, J. 2006. An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes that Lead to Payoffs in Productivity, Quality, and Risk Management. *Software Engineering, IEEE Transactions on,* 32, 433-453.

DAVENPORT, T. H. & PRUSAK, L. 1998. *Working Knowledge How Organizations Manage What They Know,* Boston, Harvard Business School Press.

DAWSON, R. 2000. *Developing Knowledge-Based Client Relationships The Future of Professional Services,* Boston, Butterwoth-Heinemann.

DE OLIVEIRA, K. M., ZLOT, F., ROCHA, A. R., TRAVASSOS, G. H., GALOTTA, C. & DE MENEZES, C. S. 2004. Domain-oriented software development environment. *Journal of Systems and Software,* 72, 145-161.

DEMARCO, T. 1979. *Structured Analysis And System Specification,* New Jersey, Prentice-Hall.

DORFMAN, M. & THAYER, R. H. 2000. *Software Requirements Engineering,* Los Alamitos, California, IEEE Computer Society Press.

EASTON, G. 2010. Critical realism in case study research. *Industrial Marketing Management,* 39, 118-128.

EL EMAM, K. & KORU, A. G. 2008. A Replicated Survey of IT Software Project Failures. *Software, IEEE,* 25, 84-90.

ELLIS, K. 2008. Business Analysis Benchmark The Impact of Business Requirements on the Success of Technology Projects. IAG Consulting
FLYVBJERG, B. 2006. Five Misunderstandings About Case-Study Research. *Qualitative Inquiry,* 12, 219-245.
FRIEDMAN, T. 2002. A Strategic Approach to Improving Data Quality. 19 June 2002 ed.: Gartner.
GANE, C. S., T. 1979. *Structured Systems Analysis: Tools and Techniques,* New Jersey, Prentice-Hall Inc.
GAUSE, D. C. & WEINBERG, G. M. 1990. *Are Your Lights On? How to Figure Out What the Problem REALLY Is,* New York, Dorset House Publishing.
GERRING, J. 2004. What Is a Case Study and What Is It Good for? *Americcan Political Science Review,* 98, 341- 354.
GHARAJEDAGHI, J. 2006. *Systems Thinking Managing Chaos and Complexity: A Platform for Designing Business Architecture* Amsterdam, Elsevier.
GLASS, R. L. 2003. Facts and fallacies of software engineering Boston: Addison-Welsey.
GOGUEN, J. A. & LINDE, C. 1993. Techniques for requirements elicitation. *In:* Requirements Engineering, 1993., Proceedings of IEEE International Symposium on, 4-6 Jan 1993 1993. 152-164.
GREENSPAN, S., MYLOPOULOS, J. & BORGIDA, A. 1994. On formal requirements modeling languages: RML revisited. *In:* Proceedings of 16th International Conference on Software Engineering, 16-21 May 1994, 1994 Los Alamitos, CA, USA. IEEE Comput. Soc. Press, 135-47.
HOFFMANN, A. & LESCHER, C. 2009. Collaboration and Intercultural Issues on Requirements:Communication, Understanding and Softskills (CIRCUS). *In:* Requirements: Communication, Understanding and Softskills, 2009 Collaboration and Intercultural Issues on, 2009. 1-4.
HOFMANN, H. F. & LEHNER, F. 2001. Requirements engineering as a success factor in software projects. *Software, IEEE,* 18, 58-66.
HOOD, C., WIEDEMANN, S., FICHTINGER, S. & PAUTZ, U. 2008. *Requirements Management The Interface Between Requirements Development and All Other Systems Engineering Processes* Springer Berlin Heidelberg.
HOOKS, I. F. & FARRY, K. A. 2001. *Customer-Centered Products Creating Successful Products Through Smart Requirements Management,* New York, AMACOM/American Management Association.
HOST, M. & RUNESON, P. 2007. Checklists for Software Engineering Case Study Research. *In:* Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on, 20-21 Sept. 2007 2007. 479-481.
HSIA, P., DAVIS, A. M. & KUNG, D. C. 1993. Status report: requirements engineering. *IEEE Software,* 10, 75-9.
IEEE 1990. IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610.12-1990,* 1.
IEEE COMPUTER SOCIETY, I. 2004. *Guide to the Software Engineering Body of Knowledge (SWEBOK),* Angela Burgess.
IIBA, I. I. O. B. A. 2009. *A Guide to the Businee Analysis Body of Knowledge (BABOK Guide),* Toronto.
JACKSON, M. 1983. *System Development,* New Jersey, Prentice Hall International.
JACKSON, M. 1995. The world and the machine. *Proceedings of the 17th international conference on Software engineering.* Seattle, Washington, United States: ACM.
JONES, C. 2007. "Chapter 1 - Introduction". Estimating Software Costs: Realism to Estimating. Second Edition ed. New York: McGraw-Hill.
KAIYA, H. & SAEKI, M. 2006. Using Domain Ontology as Domain Knowledge for Requirements Elicitation. *In:* Requirements Engineering, 14th IEEE International Conference, 2006. 189-198.
KAMATA, M. I. & TAMAI, T. 2007. How Does Requirements Quality Relate to Project Success or Failure? *In:* Requirements Engineering Conference, 2007. RE '07. 15th IEEE International, 2007. 69-78.
KAMATA, M. I., YOSHIDA, A. Y., YOSHIDA, H. & AOKI, N. 2007. Figure Out the Current Software Requirements Engineering - What Practitioners Expect to Requirements Engineering? *In:* Software Engineering Conference, 2007. APSEC 2007. 14th Asia-Pacific, 2007. 89-96.
KARLSSON, L., DAHLSTEDT, A. G., REGNELL, B., NATT OCH DAG, J. & PERSSON, A. 2007. Requirements engineering challenges in market-driven software development - An interview study with practitioners. *Information and Software Technology,* 49, 588-604.
KOTONYA, G. & SOMMERVILLE, I. 1998. *Requirements Engineering Processes and Techniques,* Chichester, John Wiley & Sons.
LABUSCHAGNE, L. & MARNEWICK, C. 2009. *The Prosperus Report 2009: IT Project Management Maturity versus Project Success in South Africa,* South Africa, Project Management South Africa.
LEDERER, A. L. & PRASAD, J. 1992. Nine management guidelines for better cost estimating. *Commun. ACM,* 35, 51-59.
LEFFINGWELL, D. & WIDRIG, D. 2000. *Managing Software Requirements A Unified Approach,* Boston, Addison-Wesley.
LEWINS, A. & SILVER, C. 2007. *Using Software in Qualitative Research A Step-by-Step Guide,* Los Angeles, SAGE.
MACNEALY, M. S. 1997. Toward better case study research. *Professional Communication, IEEE Transactions on,* 40, 182-196.
MAIDEN, N. 2008. Requirements 25 years on. *IEEE Software,* 25, 26-28.

MCMANUS, J. & WOOD-HARPER, T. 2007. Understanding the Sources of Information Systems Project Failure. *Management Services,* 51, 38-43.
NOOR, K. B. M. 2008. Case Study: A Strategic Research Methodology. *American Journal of Applied Sciences,* 5, 1602-1604.
NUSEIBEH, B. & EASTERBROOK, S. 2000. Requirements engineering: a roadmap. *Proceedings of the Conference on The Future of Software Engineering.* Limerick, Ireland: ACM.
OLIVIER, M. S. 2009. *Information Technology Research A practical guide for Computer Science and Informatics,* Pretoria, Van Schaik.
PERRY, D. E., SIM, S. E. & EASTERBROOK, S. 2005. Case Studies for Software Engineers. *In:* Software Engineering Workshop - Tutorial Notes, 2005. 29th Annual IEEE/NASA, 3 & 8 April 2005 2005. 96-159.
PFLEEGER, S. L. & ATLEE, J. M. 2006. *Software Engineering Theory And Practice*, Pearson Education International.
POLANYI, M. 1966. *The Tacit Dimension,* Chicago, The University of Chicago Press.
PRIETO-DÍAZ, R. 1990. Domain analysis: an introduction. *SIGSOFT Softw. Eng. Notes,* 15, 47-54.
RANDELL, B. & NAUR, P. 1968. Software Engineering Report on a conference sponsored by the NATO Science Committee Garmisch, Germany.
RATCHEV, S., URWIN, E., MULLER, D., PAWAR, K. S. & MOULEK, I. 2003. Knowledge based requirement engineering for one-of-a-kind complex systems. *Knowledge-Based Systems,* 16, 1-5.
REGNELL, B., KAMSTIES, E. & GERVASI, V. 2004. Summary of the 10th Anniversary Workshop on Requirements Engineering. *In:* Requirements Engineering: Foundation for Software Quality, 2004.
ROSS, D. T. & SCHOMAN, K. E., JR. 1977. Structured analysis for requirements definition. *IEEE Transactions on Software Engineering,* SE-3, 6-15.
ROYCE, W. W. 1970. Managing the development of large software systems: concepts and techniques. *In:* 1970 WESCON technical papers volume 14, Western electronic show and convention, 25-28 Aug. 1970, 1970 Los Angeles, CA, USA. WESCON, 8 pp.
RUPP, C. 2002. Requirements and psychology. *IEEE Software,* 19, 16-18.
SCHREIBER, G. 2000. Knowledge Engineering and Management: The CommonKADS Methodology. The MIT Press.
SENGE, P. M. 2006. *The Fifth Discipline: The Art and Practice of the Learning Organization* New York, Doubleday.
SIDDIQI, J. 1996. REQUIREMENT ENGINEERING: The Emerging Wisdom. *Software, IEEE,* 13, 15.
SIMONS, H. 2009. *Case Study Research in Practice,* Los Angeles, Sage.
SOMMERVILLE, I. 2001. *Software Engineering,* Harlow, Addison-Wesley.
SOMMERVILLE, I., RODDEN, T., SAWYER, P., BENTLEY, R. & TWIDALE, M. 1992. Integrating ethnography into the requirements engineering process. *In:* Proceedings of IEEE International Symposium on Requirements Engineering (Cat. No.92TH0491-1), 4-6 Jan. 1993, 1992 Los Alamitos, CA, USA. IEEE Comput. Soc. Press, 165-73.
SOMMERVILLE, I. & SAWYER, P. 1997. *Requirements Engineering A good practice guide,* Chichester, John Wiley & Sons.
STAKE, R. E. 1995. *The Art of Case Study Research,* Thousand Oaks, Sage Publications.
SUTCLIFF, A. G., ECONOMOU, A. & MARKIS, P. 1999. Tracing requirements errors to problems in the requirements engineering process. *Requirements Engineering,* 4, 134-51.
SVEIBY, K. E. 1997. *The New Organizational Wealth Managing & Measuring Knowledge-Based Assets*, San Francisco, Berrett-Koehler Publishers, Inc.
VAN DER MERWE, L. 2002. *A product development process for a photovoltaic water pump system in a small to medium enterprise.* Doctor Ingeneriae, Rand Afrikaans University.
VAN DER MOLEN, H. T., SCHMIDT, H. G. & KRUISMAN, G. 2007. Personality characteristics of engineers. *European Journal of Engineering Education,* 32, 495-501.
VAN LAMSWEERDE, A. 2009. *Requirements Engineering From Systems Goals to UML Models to Software Specification,* Chichester, John Wiley & Sons Ltd.
VERNER, J., SAMPSON, J. & CERPA, N. 2008. What factors lead to software project failure? *In:* Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on, 3-6 June 2008 2008. 71-80.
VILLER, S., BOWERS, J. & RODDEN, T. 1999. Human factors in requirements engineering:: A survey of human sciences literature relevant to the improvement of dependable systems development processes. *Interacting with Computers,* 11, 665-698.
WALIA, G. S. & CARVER, J. C. 2009. A systematic literature review to identify and classify software requirement errors. *Information and Software Technology,* 51, 1087-1109.
WASSON, C. S. 2006. *System Analysis, Design, and Development Concepts, principles, and Practices,* Hoboken, Wiley-Interscience.
WESTLAND, J. C. 2002. The cost of errors in software development: evidence from industry. *Journal of Systems and Software,* 62, 1-9.
WILLIAMS, D. & BECKER, S. March 2005. Top 10 Strategic Risk Initiatives 2005 The Clock Is Ticking. *Financial Insights,* 1.

WU, Q., YING, M., WANG, J., XIE, M., CHEN, Z. & CHEN, J. 2009. A discussion on three critical issues for assuring quality of the users' requirements specification. *In:* Industrial Engineering and Engineering Management, 2009. IE&EM '09. 16th International Conference on, 2009. 690-693.

YIN, R. K. 2009. *Case Study Research Design and Methods,* Los Angeles, Sage.

ZONG-YONG, L., ZHI-XUE, W., YING-YING, Y., YUE, W. & YING, L. 2007. Towards a Multiple Ontology Framework for Requirements Elicitation and Reuse. *In:* Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International, 2007. 189-195.

# APPENDIX A. Questions in Questionnaire

The following questionnaire was used as a third data collection approach.

Name:

Project:

Date completed:

| General feedback on individual role during project | | | | | |
|---|---|---|---|---|---|
| 1.1 | How long have you been employed by the organisation prior to be assignment on the project? | | | | |
| | | | | | |
| 1.2 | How long have you been employed by the business unit? | | | | |
| | | | | | |
| 1.3 | Rate your knowledge on the business processes and rules within the business unit. Highlight the appropriate selection and explain your selection. | | | | |
| | Extremely poor | Below average | Average | Above average | Excellent |
| | | | | | |
| 1.4 | Rate the importance of acquiring knowledge of the business processes and rules with regards to optimally perform your activities and tasks. Please elaborate on your answer? | | | | |
| | Not at all important | Slightly Important | Neutral | Somewhat Important | Very Important |
| | | | | | |
| 1.5 | What was (is) your role on this project? Please mention the different roles during the different phases of the project. | | | | |

| | |
|---|---|
| | |
| 1.6 | What was your involvement throughout the entire project life cycle i.e. consultative, fully allocated, etcetera? |
| | |
| **Requirements process followed during implementation phases** | |
| 2.1 | According, to you, how will you define requirements? |
| | |
| 2.2 | What was the process followed to gather requirements during the project life cycle? Please mention if different processes were followed during the different phases of the project. |
| | |
| 2.3 | If different processes were followed as per question 2.2, please provide reasons why different approaches were followed. |
| | |
| 2.4 | Requirements are the process of activities that are performed to understand the problem that needs a solution. Describe each of the participants that you interacted with during the requirements process. Please note whether different participants were part of the process during different phases of project. |
| | |
| 2.5 | What contribution did each participant made to the requirements process? |

| | |
|---|---|
| | |
| 2.6 | Did the requirements enable you to perform better on the project? |
| | |
| **Quality of Requirements** | |
| 3.1 | Were the requirements produced during the project life cycle of poor or good quality? Please mention whether different answers are relevant for the different phases of the project. |
| | |
| 3.2 | Explain the factors that contributed to the quality of the requirements as mentioned in question 3.1. |
| | |
| 3.3 | Is there one significant factor that stood out during the project life cycle that contributed to quality requirements ? |
| | |
| 3.4 | Did the quality of the requirements have an influence on the overall delivery of the project e.g.<br>• Resource workload<br>• Rework caused<br>• Amount of defects<br>• Other influence<br>Please elaborate. |

| | |
|---|---|
| | |
| **Customer satisfaction** | |
| 4.1 | In your opinion, were the customers satisfied with the delivered system? Motivate your answer with reasons. Please mention whether different answers are relevant for the different phases of the project. |
| | |
| 4.2 | In your opinion, was the project an overall success? |
| | |
| 4.3 | What was the main contributing factor for either the success or failure of the project? |
| | |
| **Concluding remarks** | |
| 5.1 | Please include any concluding remarks |
| | |

***