A Systems Perspective of the Operational Management of a Mobile Communications Platform.

Shaun Courtney CRTSHA001

May 30, 2008

Declaration

I hereby:

(a) grant the University free license to reproduce the above thesis in whole or in part, for the purpose of research;

(b) declare that:

(i) the above thesis is my own unaided work, both in conception and execution, and that apart from the normal guidance of my supervisor, I have received no assistance apart from that stated below;

(ii) except as stated below, neither the substance or any part of the thesis has been submitted in the past, or is being, or is to be submitted for a degree in the University or any other University.

(iii) I am now presenting the thesis for examination the thesis for examination for the Degree of MIndAdmin.

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. I have used the LaTeX Natbib style for citation and referencing.

Acknowledgements

Firstly, I would like to thank Corrinne Shaw for her patience assistance and continued support throughout the degree. Secondly, I would like to thank my family for the many hours they have allowed me to work on this degree. Thirdly, I would like to thank my wife Jenni for her support and love.

Contents

List of Figures

List of Tables

List of Acronyms

BOT Behavior Over Time.

e-business electronic business, especially utilizing the Internet.

ICT Information and Communication Technology.

IS Information Systems.

IT Information Technology.

IVR Interactive Voice Response.

OOP Object Orientated Programming.

RD Root Definition.

ROI Return On Investment.

RUP Rational Unified Process.

SMPP Short message peer-to-peer protocol.

SMS Short Message Service.

SMSC Short Message Service Centre.

SMTP Simple Mail Transfer Protocol.

SSM Soft Systems Methodology.

UML Unified Modeling Language.

WASP Wireless Access Service Provider.

XP Extreme Programming.

Abstract

Since the Dot-Com crash during 2000 (Ljungqvist and Wilhelm Jr., 2003), Information Technology (IT) companies, which prior to this were given *carte blanche*, have been severely reigned in. According to Drobik (2001, pp 1) the key issue facing IT providers today is "How will e-business adoption improve operational performance, increase revenue, reduce expenses and accelerate earnings growth?" This dissertation seeks to address these types of questions, looking specifically at a company that provides the e-business solution within the mobile communication space. In this study the product provided was a mobile communications platform that was used to transmit information via a wide variety of methods and protocols. The research question for this thesis was ""How does the lack of knowledge about the management of the software engineering methodology, employed by the management team and deployed by the developers, affect the operational deployment costs of the platform?" In the context of operational management this was defined as a messy problem, according to Dick (1997) Action Research is a cyclical process where both change and learning can be undertaken simultaneously and can be used to address messy management problems. Within the action research paradigm, Soft Systems Methodology was determined the most suitable methodology to address the research question. The result of the study determined that by controlling two factors, firstly the elements *within* the platform (the software engineering development) and secondly the environment *without* the platform the hosting environment would have significant consequences in reducing the costs of the deployment and management of the platform. This being said, in general the knowledge gained in resolving the research question for this study can be applied to this class of Information Technology problems.

Chapter 1

Introduction and Problem Statement

A spectacular example of a software systems failure, described by Johnson (2005) and Jackson (2006), was the automated baggage handling system of Denver International Airport, which is one of the largest international airports in the world, covering over 53 square miles. A network of conveyors 26 miles long was intended to autonomously route baggage around the airport with pin point accuracy and so reduce the waiting time for air travelers. From its inception the $186 million dollar project was doomed. Originally due to open 31 October 1993 it was delayed and eventually opened on the 28 February 1995. The opening had been delayed by over 16 months. The time delay was ascribed to “project overrun” which cost hundreds of million dollars, at a rate of one million dollars a day. Despite years of adjustments and amendments, the system never ran reliably and resulted in the liquidation of the company that designed the system: BAE Automated Systems (who at the time were responsible for 90% of the USA baggage handling); not to mention it contributed to the bankruptcy of its principle user, United Airlines. In response to increased financial pressure the airports general manager for customer service in Denver, Jim Kyte, decided in 2005 to pull the plug on the baggage system. Quoted in the New York Times he said "We’re going back to the future," referring to reverting to manual baggage processing. This decision resulted, in maintenance alone, in a saving of about $1 million dollars a month (Johnson, 27 August 2005). The failure of this system highlighted an aspect of the price paid by poor software design and implementation. On closer examination of this automated baggage handling

system a number of systemic issues were identified.

Since the Dot-Com crash during 2000 (Ljungqvist and Wilhelm Jr., 2003), Information Technology (IT) companies, which prior to this were given *carte blanche* over their budgets, have been severely reigned in. These companies are no longer run by computer visionaries, but rather by accountants, CEO's and Boards of Executors, this indicates that the nature of the playing field has changed.

According to Drobik (2001, pp 1) the key issue facing IT providers today is "How will e-business adoption improve operational performance, increase revenue, reduce expenses and accelerate earnings growth?" This dissertation seeks to address these types of questions, looking specifically at a company that provides the e-business solution within the mobile communication space. In this study the product provided was a mobile communications platform that was used to transmit information via a wide variety of methods and protocols. The primary method used to transmit information was via mobile Short Message Service (SMS), but with the capability of transmission via facsimile, voice, web and email as well.

1.1 Overview of this Thesis

At this stage an overview of the thesis is provided, not only to provide an outline of the document but to clearly present the intention of this thesis so as to inform the reader of the direction of the project.

The theory used to consider I.T. management problems, such as those described previously, have been discussed in Chapter 1. A practical approach to problem solving is Action Research described in section 1.2, Action Research involves an iterative learning cycle, allowing management to learn and adapt at each stage of a project. In section 1.3 the formulation of the problem has been considered and the work of Ackoff (1981), who defined problems within a complex environment as messy problems, has been reviewed. Ackoff (1981) went on to consider the different approaches to solving these problems: the clinical, research and design approaches. Because the design approach was most relevant to this thesis it was considered in more detail. It is a participative process that is adaptive through time and therefore appropriate to the thesis. Finally the practical problem: "The deployment of a mobile communication platform in multiple and varying sites globally" was discussed in section 1.4.

Following on Chapter's 2 through 4 provide the background detail and theory as well as the frameworks and methodologies used in this study relevant to the case study presented in Chapter 5.

In Chapter 2, "Software Engineering and Operation Management", the evolution of software development was explored ranging from the Waterfall method, through Agile Development and finally to the Rational Unified Process which was the method that was implemented for this thesis' application, because of its iterative nature and the associated advantages to both developers and clients.

Chapter 3, "Philosophical and Research Framework", considered the philosophy behind the search for knowledge in the field of social research. Basically, the search for knowledge can be reduced to: Experience, Reasoning and Research (Cohen, Manion, and Morrison, 2000). The research factor was explored more deeply and a four-layered research framework was defined not only theoretically but in terms of this thesis and the practical problem as well.

The final Chapter that provides background theory necessary to the application, is Chapter 4: the Action Research Methodology selected; Soft System Methodology (SSM) is described. The methodology was designed with the specific role of dealing with change and messy problems such as the practical problem presented in the following Chapter. The SSM process produced a conceptual model of the system in focus and this model was then used to formulate the steps that needed to be undertaken in the "real world". The outcome of these actions were then evaluated against the models expected outcomes.

In Chapter 5 the practical problem and thus the research question were addressed with regard to the deployment of mobile communication platform. Three iterations of SSM were undertaken, the outputs of the previous cycle were fed into the system to refine the conceptual model. This model was then utilized to predict the outcome of further interventions and so converged toward an answer to the research question. This led to a better understanding of the costs involved in the management of the methodology used by the developers and the development and testing process were managed differently which led to a measurable reduction in deployment costs in terms of manpower and time and also resulted in a more stable platform which proved to be easier to manage. For the clients this resulted in a better product that matched expectations and performed accordingly.

In the final Chapter, Chapter 6, the practical problem is reviewed with respect to the research question. The process of the application of SSM (within an Action

Research paradigm) to the process of deployment of a mobile communicating platform is reviewed. This review demonstrates that SSM and Action Research are appropriate to addressing this type of messy management problem in an IT environment.

1.2 Action Research

Action Research is a practical approach to problem solving. According to Dick (1997) Action Research is a process where both change and learning can be undertaken simultaneously. This process is cyclical and consists of three phases.

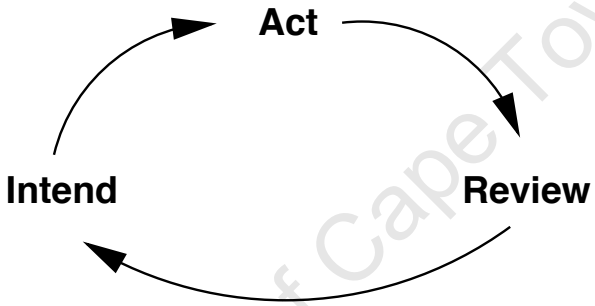


Figure 1.1: Action Research (Dick, 1997)

In the first phase, the researcher intends to do something or make some changes, these changes are acted upon in the second phase and subsequently in the third phase they are reviewed. These phases are illustrated in figure 1.1.

In the following figure, figure 1.2, an expanded model of Action Research is described as the Research Process. This process is described in two stages, which are outlined here: the first stage is concerned with identifying the practical problem. This practical problem does not arise in a vacuum, but rather occurs as the result of a particular concern which is part of a larger situation. In the second stage in order to solve the practical problem, it needs to be framed as a research question. The research question leads to a research problem. A research problem is one which is defined in terms of a lack of knowledge in a particular area (Booth et al., 1995). With action research an answer to this question is found. This answer leads to new knowledge, which is "actionable". This action leads to the practical problem being addressed which in turn addresses the particular concern. This two stage method of solving the practical problem was used as the conceptual basis for this study.

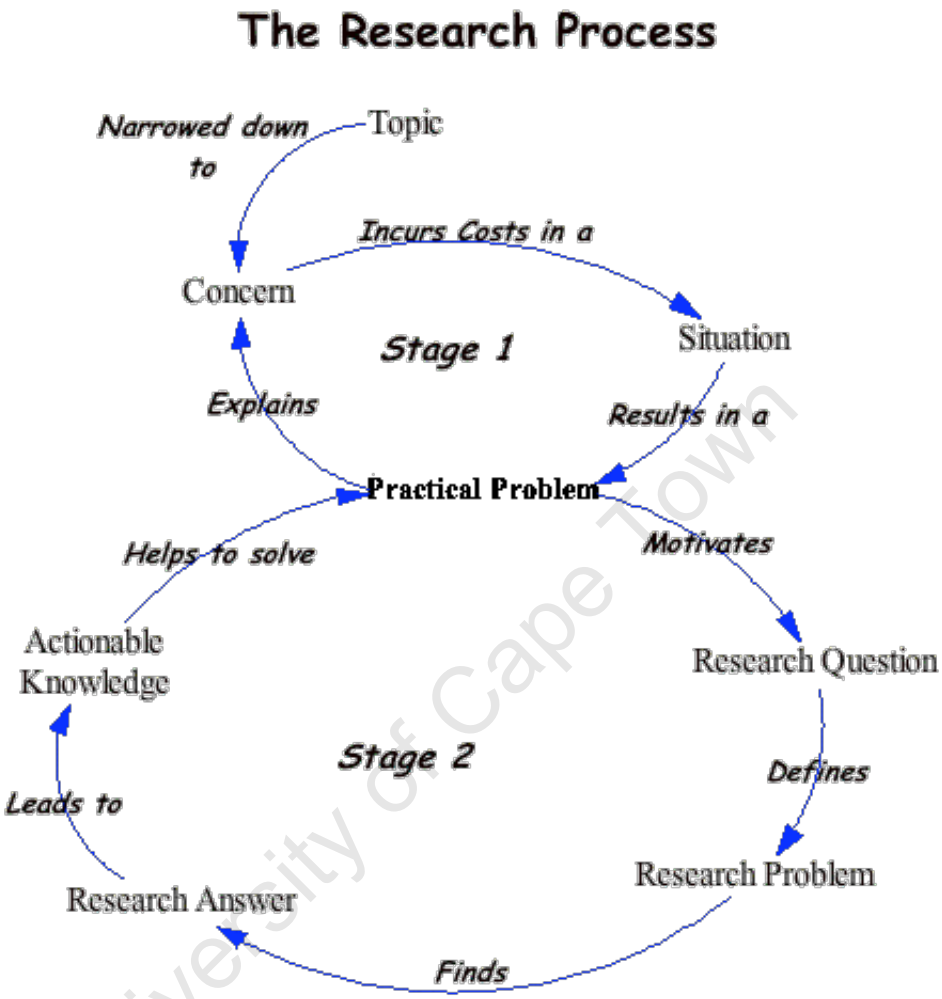


Figure 1.2: The Research Process (Ryan, 2005)

Ryan (2005) argued that management is about keeping the variables of the system within limits. A managers task is to utilize the resources at their disposal to keep their systems variables within check. Once these variables move beyond acceptable limits within the system, it can be described as a practical problem. From a systems perspective these behaviours can be graphed in a Behaviour over Time graph. To sketch one of these graphs one would select a time period, on the horizontal axis and plot the variable of interest over time. Any unacceptable deviance from the desired mean indicates a problem that needs to be addressed.

Research in this type of management practice is applied research. The applied research addresses the problem that has arisen in a situation by formulating it as a practical problem. This is done by defining the problem as a theory or hypothesis. It then attempts to identify the elements of the problem: this is the research question and to explain how the behavior can be explained by these elements. To do so leads to the research problem, defined by Booth et al. (1995) as "always some version of not knowing or not understanding something" (pp 62). By addressing this research problem one is able to construct a theory as to what is happening within this system. This theory forms the basis of actionable knowledge and is then acted upon. By implementing the action plan one helps solve the practical problem. This action is reflected upon and the cycle repeats until the practical problem is addressed. Figure 1.3 demonstrated, again, stage 2 of the research process which was described previously in figure 1.2.

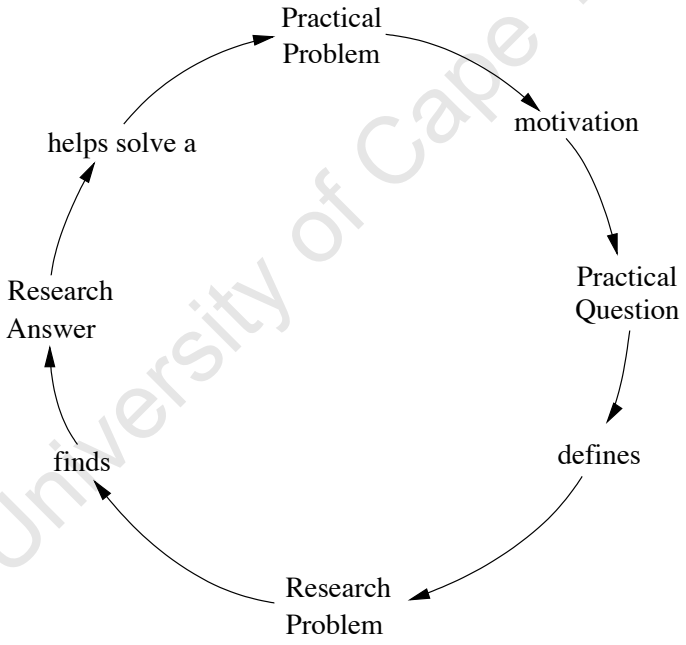


Figure 1.3: The relationship between practical and research problems. From Booth et al. (1995)

1.3 Problem Formulation

Albert Einstein said, "We can't solve problems by using the same kind of thinking we used when we created them." This is clearly demonstrated by the application

of Action Research to practical problems, particularly in social studies. Because problems do not occur in isolation, they can be seen as part of a larger system that is constantly evolving. The result is that we see large interacting systems each with their own problems. Now, Ackoff (1981) calls these types of problems messy problems and proposes a number of methods to define and deal with them. The reason messy problems are addressed in this section and thesis is because in management systems the nature of problems tend not to occur in neat silos, but rather they are interspersed with other issues from the surrounding environment and are thus, by definition, messy problems.

Ackoff (1981) defined a problem as a situation that satisfies three conditions:

> First, a decision-making individual or group has alternative course of action available: second, the choice made can have a significant effect; and third, the decision maker has some doubt as to which alternative should be selected." (Ackoff, 1981, pp. 20).

Ackoff noted that there are three outcomes or solutions to these problems: they can be resolved, solved or dissolved:

To resolve a problem one selects a course of action whose outcome is such that it satisfies or is sufficient to resolve the problem. The research that meets this criteria is generally qualitative and is based on surveys of opinions or attitudes. The methods employed are generally questionnaires or interviews. Once most stakeholders are content with the solution that satisfies the problem it is considered resolved.

To solve a problem is to select a course of action that is believed to result in the best possible outcome. This can be seen as optimizing the solution, as it relies on the scientific method of solving problems and it is based on an objective philosophy. The result of this approach is the clinical treatment of qualitative data as this type of data does not fit well into their models, but this is preferred to a more comprehensively formulated problem with a less optimal solution (Ackoff, 1981).

To dissolve a problem is to modify "the nature, and/or environment, of the entity in which it is embedded so as to remove the problem" (Ackoff, 1981). Those who follow such a method tend to idealize the problem. The reason is that they tend to modify the system involved and bring it into the desired state. As a result of this approach the dissolver uses both the "resolve" and "solve" method to bear on the

problem space. Using both tools and methods from each of the previous solutions spaces to modify the system in which the problem occurs. The result of this type of solution is development.

Ackoff (1981) noted that few problems are ever permanently resolved, solved or dissolved, but of all the methods employed, problems that are dissolved tend to have longer lives. Ackoff (1981) suggested that the best method to approach messes is via the management of the messes themselves rather than the short term solution of the problems. To do this mess management requires planning rather than problem solving.

Ackoff (1981) said there are three broad approaches to mess management and these are patterned after the three methods of solving problems. So to resolve a problem corresponds to the clinical approach, to solve a problem corresponds to the research approach and finally to solve a problem corresponds to the design approach. These three approaches are explained further here.

In the Clinical Approach to resolving problems the approach to managing the mess considers the whole system but does not seek to analyze it. Clinical planners use methodology that is participative in nature and the clinician serves chiefly as convener. The participants are involved in activities that result in consensus on what the mess is and how to tackle it. The methods employed are those frequently used in the behavioral sciences and involve participation between the various stakeholders. Although comprehensive in its approach to planning, the end result lacks definitiveness as there is no explicit criteria to evaluate the results. The result of removing what is unwanted from the mess does not result in what is wanted.

The Research Approach, patterned after the "solve process" outlined above, begins up front by studying the mess environment and tries to identify their parts and interrelations. Using a reductionist framework, components are isolated and examined. The components are processed using traditional problem-solving methods. The results are however far from satisfactory, because the system has been decomposed it loses its essential qualities and properties and hence the results are not a good fit with the system.

The Design Approach, mirrors the dissolving method outlined before and is a synthesis of the clinical and research approach. The aim is to use the advantages of the two systems and avoid the weaknesses. This approach to mess management consists of five phases, which were outlined by Ackoff (1981) as:

Formulating the mess In this phase the systemic properties of the mess are captured and highlighted. The aim is to capture the future of the system as it would be if the system continued unmodified.

Ends planning This involves choosing the desired outcome, ideals, objectives and goals that the relevant stake-holders would ideally like to change in the idealized system defined in the previous phase if they were able to do so today. This process defines the gap between what the system is and what it ought to be.

Means planning Now that the gaps have been identified the methods for filling these gaps can be identified.

Resource planning In this phase the gaps and means to fill them have been identified, as well as the resources needed to satisfy these requirements.

Design of implementation and control. The final phase is the design the implementation given the resources to control the mess.

The five phases employed in this approach are participative and all stake-holders can be involved in the process. This process is designed to be adaptive and can be used for continuous organizational improvement.

1.4 Defining the Practical Problem.

The purpose for considering the background theory in the previous section was to apply this knowledge to a real world or practical problem. The concern in this study was the deployment of a mobile communications platform in multiple and varying sites globally, before proceeding a description of the Mobile Communication Platform is required.

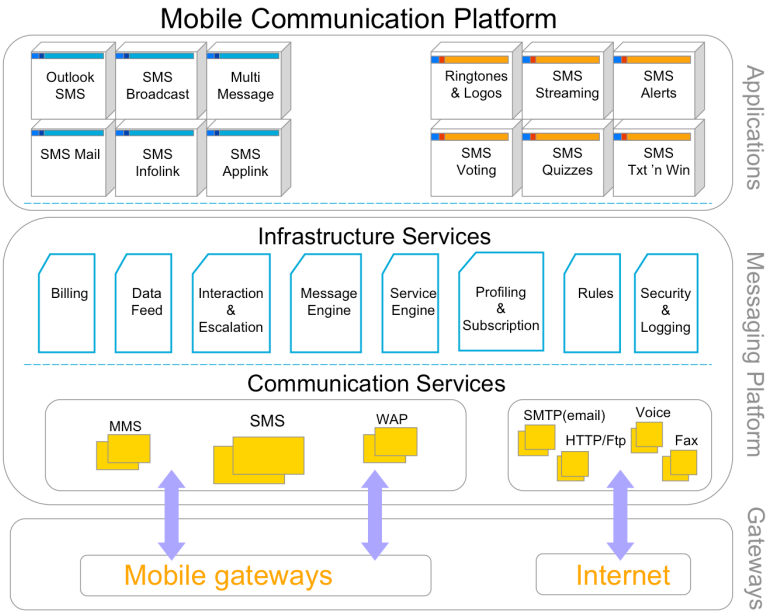


Figure 1.4: The mobile communication platform

The mobile communication platform is a collection of software programs that can be used to transfer digital information from one party to another, as illustrated in figure 1.4. The channel of communication is input agnostic, which means the input and output information need not have the same format. This is shown in figure 1.4 as the Communication Services. A simple example would be email to sms, such as email sent to the platform can be to the sent via SMS out to one or more recipients. The platform consists at its core of a collection of software modules that perform fundamental services such as billing, collecting and processing input and output as well as the business logic of the system, in figure 1.4 these are referred to as the Infrastructure Services. The Infrastructure Services in turn communicate with the various gateways which are used to deliver output and receive input to the system. On top of the messaging platform sits the client facing applications which provides services to the users of the system. In figure 1.4 this is shown in the three broad levels. This level with which the client interacts the is the Application level. Here the email to SMS example was described, the communication in the form of email passes via the SMTP gateway into the Messaging Platform where it passes via a variety of Infrastructure Services engines to be processed and finally passed to the

SMS gateway to be sent out via SMS.

The mobile platform can be used in a variety of configurations. Firstly the clients can use one of the hosted services such as email to sms. The platform is fully hosted, maintained, configured and operated by the development company. A second method of using the platform is to co-locate a hosted server which is dedicated to the client company. The service is fully maintained and managed by the developers and the client just runs their service on this dedicated platform. This method is used by companies that require a customized platform but do not have the necessary skills to maintain the software or hardware. The third option is to purchase a customized platform that is installed and maintained by the development company, but hosted on the clients premises. This is an option that telecommunication companies require. They need the server hosted in their data centers but do not want to maintain and operate the platform. The last method of running the platform is for the client to purchase the platform outright and host and maintain it in their data center themselves. This method is preferred by clients such as financial institutions.

From the management perspective the underlying question asked by the operational team and ultimately by the company itself was: How does the company deploying these solutions improve its operational performance, increase revenue and reduce expenses in order to accelerate earnings growth? Framed more specifically from the operations perspective : "What elements were contributing to the operational and deployment costs of the mobile communication platform?" Reducing costs and increasing profit margins was the practical problem. In order to address this problem one needs to know what contributes to rising costs. To do this the cost factors needed to be identified and then managed so as to reduce overall costs and increase profits.

In this thesis the concern was expressed conceptually as a Behavior Over Time graph (schematically illustrated in figure 1.5). This figure shows the variable of the profit margin, over time. Through time some defined level of profit is an acceptable return on investment (ROI), but values below the pre-selected threshold are not acceptable, in this example the threshold is defined as the break even costs. This may still be profitable but not a desirable ROI as defined by the company. Management is chiefly concerned with regulation, that is keeping essential variables to the business within acceptable limits. These limits are defined externally either by the CEO or the market. When these variables move beyond acceptable limits, it is

the job of the manager to correct these fluctuations and bring the system back into acceptable balance within an acceptable boundaries.

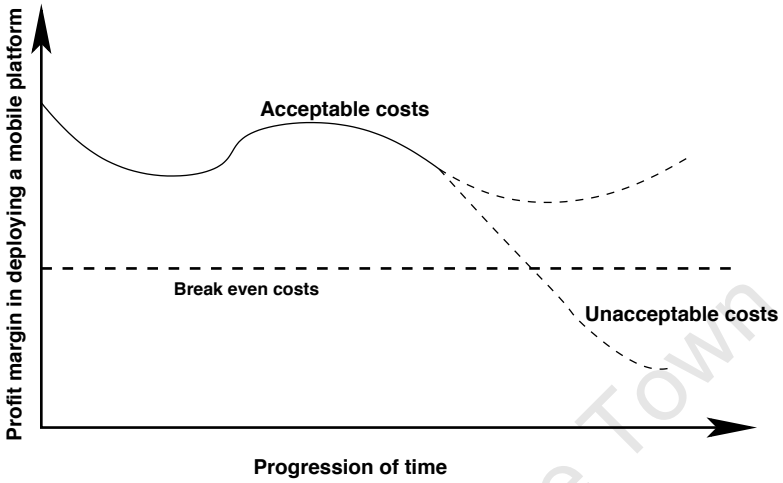


Figure 1.5: Behaviour over time graph showing the potential profitability of deployment of a platform over time

The concern, in this study, was the cost of deploying the platform for the company in a multitude of environments. The concerns regarding deployment were heightened when the platform was installed internationally and the costs of delays in implementation incurred were obviously greater to the company than during local installation delays. These costs where in terms of manpower on the development and deployment teams, as well as costs in terms of the quality of the platform deployed. Spending a week installing the platform with constant patches did not install confidence in the system for the client who had just purchased the platform. Should the environment not be correctly identified and its constraints not been clearly defined when deployed, the project would result in overruns. It was a concern to the clients that a team of engineers were required to install and configure the product on the clients premises.

In this thesis the management of the deployment of a mobile communications platform, which was installed at four sites during 2001 to 2004, was considered in order to demonstrate that the success depends largely on the management around the software engineering method used. According to Lai (2001) improving the effectiveness of software development is one of the top concerns of Information Technology management. In the area of global software development and the deployment of

these projects the lack of understanding in this area results in IT projects that overrun budget, exceed allocated time and are of poor quality (Ewusi-Mensah, 1997). Akmanligila and Palviab (2004) showed that in the literature eight global development strategies have been noted and they developed a ninth but conceded that choosing a strategy was difficult because there are so many strategies available. In seeking to understand some of the root causes of these issues, the evolution of the management of the development/deployment model has been demonstrated and how the involvement of all stakeholders resulted in the decrease in costs, both in terms of manpower and time and resulted in an increase in the quality of the software deployed over time.

The practical problem in this thesis was not well defined and fell into the class of problems known as messy problems, which were described and defined earlier. Although the practical problem was ill defined it does not mean to suggest that it was not tractable. Soft Systems Methodology was chosen as the methodology to address these problems. Using the Soft Systems Methodology to solve this class of problem is common in Information Technology. Rose (2002, pp 242) best summarised the relationship between Information Systems, SSM and action research in his paper "Interaction, transformation and information systems development - an extended application of Soft Systems Methodology." He claimed that:

> "Soft Systems Methodology (SSM) has long been involved in information systems developmentthrough the medium of action research. It social constructivist paradigm and managerial focus distinguish it from most software engineering development approaches."

The aim of the study was to choose and develop a high level development/deployment management strategy and a software engineering methodology that addressed the concerns of budget and time overruns and contributed towards developing a better quality software. The goal was to help the developers shift from customization of software for each installation to designing quality software that had the necessary requisite variety to be configured in different environments.

The practical problem was determined as: How to reduce the costs associated with the development and deployment of the mobile communication platform. According to Crotty (1998) a research problem always concerns a lack of knowledge. In this study the research question is stated as "What elements do I need to better understand, in order to determine the issues involved in the development/deployment

process that are effecting cost?" To answer this question a better understanding of the variables that were contributing to the increase in the cost of the installation process was necessary. The variables needed to be of such a nature as to be able to be manipulated to address the practical problem. A rich picture was drawn and using Soft Systems Methodogy the root cause of these issues emerged. A rich picture is a diagram that depicts a messy problem graphically and attempts to identify all the stakeholders and the interconnections between them. These root causes gave a better understanding of what contributed to the high costs of the installation.

The research question was revised as, "How does the lack of knowledge about the management of the software engineering methodology employed by the management team and deployed by the developers affect the operational deployment costs of the platform?"

From an operations management perspective the question can be restated as, "How does the software engineering methodology chosen by management and employed by developers impact on the costs of deployment and what possible changes can be introduced so as to reduce the costs of deployment and management of the platform?" With highly specialized and configurable software the software engineering methodology chosen appear to have a great impact on both the reliability and maintainability of the platforms being deployed. Not fully understanding the management process of the software engineering methodology employed operationally was dooming the project to high costs and failure to deliver on customer expectations.

In summary, the practical problem was determined to be a messy problem that involved "Reducing costs and increasing profit margin in the deployment of a mobile communication platform." That being said, the aim of the study was defined as : "Choose and develop a high level development management strategy." Further examination of these questions led to the emergence of the research question: "How does the lack of knowledge about the management of the software engineering methodology employed by the management team and the developers affect the operational deployment costs of the platform?"

Chapter 2

Software Engineering and Operations Management

In this chapter software engineering has been defined and the application of those aspects of the definition that pertain to this thesis were considered. Furthermore, the role of software engineering in the design and operation of a software systems project were examined.

Firstly, the question: what is software engineering, needed to be addressed.

> "The IEEE Computer Society defines software engineering as (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1). Abran et al. (2004, pp 1-1)"

Working from this definition the important areas, which were the focus of this study, could be highlighted. These areas are development, operation and maintenance of software. The definition has been considered along these lines, the development, operation and maintenance, as they were the areas of control and management within the company concerned.

Arguably, the first and most likely well-known book on software engineering is Brooks' book "The Mythical Man-Month". Brooks Law states that adding manpower to a late software project makes it later (Brooks, 1995). The corollary of this is the more people you need to communicate with within a project the longer

it takes to get that project done. Brooks (1987, pp 10) claimed that "there is no single development, in either technology or in management technique, that by itself promises even one order-of-magnitude improvement in productivity, reliability, or simplicity." This statement raised great debate at the time of publication, but in time it proved to be correct.

A number of models for software development have emerged over time, each pertaining to its own philosophy of development and thus each having a different emphasis of priorities, not to mention a different relationship between the clients and the development teams. One of the first of these was the Waterfall Method, which was dependent on the procedural software available at that time. During the 1990s a new methodology emerged, known as Agile Software Development. Agile Software Development is better suited to the present Software Engineering environment, that evolved from the procedural to an object orientated environment as the object orientated methodologies became available. A third method discussed is that of the Rational Unified Process, which has been developed from best practices over the last few years. The Waterfall, Agile Development and Rational Unified Process methods have been discussed in this section of the thesis.

2.1 Waterfall Method

The Waterfall Method of software development was outlined by Sommerville (2001). This method is the classic development model from the late 1960s and 1970s, and still finds some proponents today. It consists of five stages cascading from one stage to the other as shown in Figure 2.1. The five stages are: analysis; design; implementation; testing and deployment. Sometimes the Waterfall Method is expanded to seven stages by splitting the implementation stage into construction and integration and the last stage of deployment into installation and maintenance. Ironically the chief argument against this model, is also the chief argument for it. That is all the specifications for the system are done up-front in the initial step. Those who support the method point to the fact that time spent up-front carefully designing the system and capturing the requirement pays dividends in the later stages. Those who criticize this method feel it is nearly impossible to get each stage perfect before moving on. Also, the nature of the software system environment is that customers requirements change and this necessitates the re-running of the model each time, from scratch, in order to accommodate the new requirements.

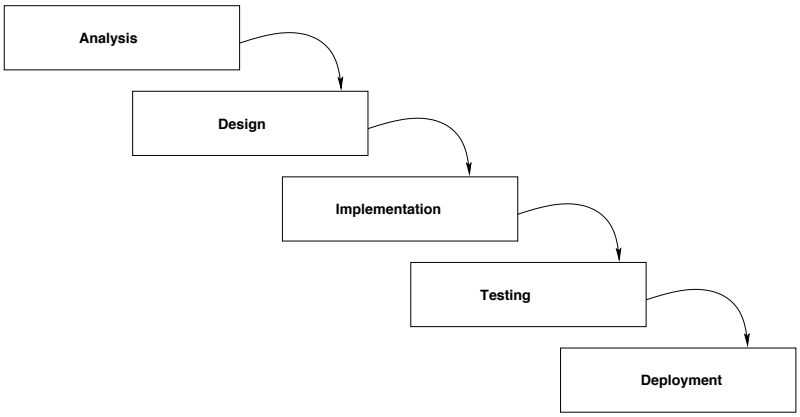


Figure 2.1: The Waterfall Method

According to Fowler and Highsmith (2001) the main emphasis of a project using this methodology was to concentrate on the processes and tools required for the project and comprehensive documentation. Initially a large portion of the project would have to be devoted to contract negotiation and subsequently following the plan according to the initial negotiation. This methodology works well for small projects but as time progresses and projects expand so it becomes progressively harder to include new features to the system and the "quick fixes" to the original code becomes too overwhelming to contain. These projects are, by definition, resistant to change and are thus only useful for smaller short term projects.

2.2 Agile Development

In 2001, a group of "organizational anarchists" met to find a new methodology for software development. They were primarily concerned with developing better software. Their focus was on individuals and interacting with them, producing working software and they valued the relationship with their clients highly and welcomed the changing requirements of the clients as the project evolved. The purpose of the Agile Methodology can be summed up in the "Manifesto for Agile Software Development" defined in Fowler and Highsmith (2001) as:

> "We are uncovering better ways of developing software by doing it and helping others to do it. We value: Individuals and interactions over

> processes and tools. Working software over comprehensive documentation Customer collaboration over contract negotiation. Responding to change over following a plan"(Beck et al., 2001)

The principles of the Agile methodology, which are commonly accepted as the underlying philosophy behind the Agile development method are listed as follows, after Fowler and Highsmith (2001):

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter time scale.
- Business people and developers work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity, the art of maximizing the amount of work not done, is essential.
- The best architectures, requirements and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tune and adjust its behavior accordingly.

The obvious advantage to the Agile methodology is that it is dynamic and relies on a changing environment in order to adapt and to work well. In order to process an Agile project one has to follow an iterative process, each new development relies on the previous code being tried and tested. As the development process occurs it is important to consider what worked well, what needs improvement and what needs more careful consideration (Fowler and Highsmith, 2001). This means that the client and development team have to maintain their working relationship throughout the project as they need to consult and adjust after each new development or improvement before continuing with the next iteration. An advantage of being involved at every iterative step for the client is that they learn which aspects of their project are viable and which are not, which are worth pursuing and which are not. Also, being involved in the process enables them to have a good understanding of the system and thus become knowledgeable users.

According to Fowler and Highsmith (2001) a number of methods adhere to the programming Agile methodology for example: Extreme Programming (XP) (Beck, 1999), Scrum, Crystal, Context Driven Testing, Lean Development and (Rational) Unified Process (see Section 2.3). The most prominent of these is Extreme Programming (Beck, 1999). Extreme programming arose in a changing Information Technology landscape where an object oriented paradigm replaced the procedural paradigm with the rise of languages like C ++ and Java as well as the birth of the World Wide Web where time to market was of crucial importance.

Extreme programming (Beck, 1999) is based on four values: **simplicity**, by using the simplest possible code new features can be added to the system more easily further into the project; **communication**, in order to avoid problems and errors that arise without it; **feedback** in order to ensure that the project doesn't wander away from its main objective and **courage** in order to tackle the large adaptions required by this process. According to Marchesi (2005) these four values imply a fifth, that of **respect** in order to achieve the high level of communication required for the process.

Unlike the waterfall method, where requirements were fixed at stage one, this model is iterative and hence able to adapt to changes in the requirements. This approach to software design allows the construction stage to be concurrent with the other development stages such as requirements, design and planning. It is highly iterative in nature and relies on delivering code in short time-scales, for the duration of the project. It is quite a revolutionary approach to software development and

the methodology itself has not stood still, but evolved over time. Despite its controversial methods it remains a good choice for a number of Software Engineering projects.

2.3 The Rational Unified Process (RUP)

The Rational Unified Process (RUP) is a software engineering process product, which was developed by a team of developers, whose chief architect was Kruchten (2003) and is available from IBM or through the Internet. It consists of many of the best practices used in software development today such as: develop software iteratively; manage requirements; use component-based architectures; visually model software; continuously model software; continuously verify software quality and to control changes to software (Kruchten, 2003). Of particular relevance to this thesis was the iterative software development process upon which this research was founded. Unlike the more formal waterfall methods this method is not a prescriptive process, but rather an adaptable process framework that can and should be adapted to each project to which it is applied. Although a propriety product exists, formed initially by Rational Software and now associated with IBM, the success of the method has led to a generic version of the processes.

In order to understand the real benefit of this model, one needs to first understand its design background. Kruchten (2003) in discussing the "Symptoms and Root Causes of Software Development Problems," he identified the characteristics of failed software projects and in so doing attempted to identify the root cause of these failures. The purpose of this exercise was to examine already existing solutions to these problems. Kruchten (2003) stated the following failures in the software development process:

- Inaccurate understanding of end-user needs
- Inability to deal with changing requirements
- Modules that don't fit together
- Software that's hard to maintain or extend
- Late discovery of serious project flaws
- Poor software quality

- Unacceptable software performance
- Team members in each other's way, making it impossible to reconstruct who changed what, when, where, and why
- An untrustworthy build-and-release process

With respect to project failures Kruchten (2003) identified the following failures:

- Ad hoc requirements management
- Ambiguous and imprecise communication
- Brittle architectures
- Overwhelming complexity
- Undetected inconsistencies in requirements, designs, and implementations
- Insufficient testing
- Subjective project status assessment
- Failure to attack risk
- Uncontrolled change propagation
- Insufficient automation

In software engineering, as in other disciplines of engineering, there is seldom a single cause for failure. By addressing these common root causes for failure Kruchten (2003) was able to use "best practices" to formulate a methodology that attempted to address the issues. Kruchten (2003) outlined the best practices used for software development and used the following six points as the basis for the new framework.

Figure 2.2: The iterative and incremental process of RUP. (The Rational Edge, 2003)

1. **Develop software iteratively**: One needs to recognize that due to a number of factors, which include the complexity of the task, the time it takes to implement and Moore's Law, that over time things will change. Moore's Law states that the number of transistors on a chip doubles about every two years (Moore, 1965). Unlike the Waterfall methods seen in section 2.1 RUP embraced change of requirements during the development phase. Change is brought about by user or client requests and changes to hardware architecture or vendor change. This process in iterative, with each new release seen and commented on by the client, illustrated in figure 2.2. With each iteration of the project a functional, although limited, project is released which helps give the client feedback and helps the developer track the risks and priorities associated with the project. Doing things this way results in the integration of new requirements in a more cost effective and less complex manner. Risks can also be detected early on and can be prioritized and tackled. Overall the product is also improved, with each cycle as the project is scrutinized.

2. **Manage requirements**: Although open to changes, it is better to have a well defined product specification up front. In order to manage the requirements

RUP is concerned with identifying and specifying what these requirements are. This has two results, one it gives the system architect the correct information to design a product that meets the clients needs and secondly, it can be used to identify what needs to be modified if the client requires changes at a later stage of the production cycle. The RUP suggests a number of steps that can be taken to generate a specification. These are step by step activities that are designed to minimize the root causes of failure and range from understanding the clients needs to managing the scope of the systems and setting up a process for change management.

3. **Use component-based architectures:** One of the buzz technologies in software development is Object-Oriented Programming (OOP). RUP encourages the use of well defined, extensible and reusable components. These map closely to the set of objects in OOP. By using components a complex system can not only be built, but each component itself can be well tested as long as the interface to the large system is maintained, even completely modified or improved without negatively impacting on the system as a whole.

4. **Visually model software:** In order to abstract the details of a complex system RUP recommends that you visually, and graphically model your software design. This can be done with the Unified Modeling Language (UML). The advantage of this method it that it abstracts the details of the code from the requirement of the system. Stepping back from the implementation directly, a team of developers can then best identify how to code the system in such a way so that it takes cognizance of the interrelationship between the different components.

5. **Continuously verify software quality:** Quality is not something that is added to the system afterwards. It needs to be designed in to the system and developed in parallel. In RUP it is assumed that each person is doing quality control and assessment all the steps of the way. In order to achieve this test, cases that meet an expected level of quality should be provided.

6. **Control changes to software:** By the very nature of the development model, one needs to cope which change. RUP defines change control methods that track, control and manage these and unexpected changes. The process is helped by the component based architecture approach and the rigorous software quality control that is inbuilt. Further the concept of secure workspaces

ensures the developers are able to effect changes without impacting on the system as a whole.

The RUP project life-cycle consists of four phases:

1. **The inception phase:** which includes the business context and financial forecast for the entire project. A number of criteria, natural to any business plan, are considered at this stage - for example, cost versus schedule estimates and actual versus planned expenditure. At this stage it is essential that a common understanding between the developers and the clients, regarding the project, is established.

2. **The elaboration phase**: where the domain analysis of the particular project is performed and the basic outline of the project is established in the form of a development plan. Here one must ensure that the software fulfills the tasks and functions outlined in the requirements of the previous phase. At the end of this phase the software architecture is in place and executable and beyond this phase changes become detrimental to the system.

3. **The construction phase:** During this stage the project becomes focused on the development of the system components. Most of the coding takes place at this stage and the project is split into multiple small, manageable components. Also the components need to be tested as units and verification that the requirements have been met correctly needs to be done at this stage.

4. **The transition phase:** The emphasis shifts from the developers to the client during this phase and includes training the client in operations and maintenance. Also if the project does not meet the requirements set out in the inception phase then the cycle repeats itself, in contrast if the requirements have been met the product release milestone is met and the development cycle ends.

Kruchten (2003) described the advantages of the RUP and main advantages of the iterative process have been discussed here. The iterative methodology associated with the Rational Unified Process has many advantages over previous methodologies; for instance it welcomes changing project requirements, which in the past would have led to late delivery, annoyed clients and frustrated developers. The

RUP allows management to assess the project in smaller "time-bites", as the financial status, the holistic product and project schedule can be tracked at each iterative step of the project. Also the nature of the process is that the lengthy development phase, associated with previous methods, when outcome was unknown until the end of development, is now broken down into fewer smaller integrations and so problems and risks that arise during the development can be addressed immediately. Addressing problems in the early phase of the project ensures that costly delays and expensive system errors can be avoided towards the end of the project. This methodology also provides robust architecture because emerging errors are corrected over a series of iterations, which in itself, increases the usefulness of the code modules; which can be developed further, which should be reused within the system and which should be bought.

From a management and business perspective this methodology is useful since all employees, from developers to testers are busy with the project from the start, as opposed to previous works where testers would be waiting for developers to finish and developers would sit and wait for the testers to finish. Many project managers avoid the RUP because they feel that the iterative process makes the project gain an endless life of its own as new modules are "hacked" onto old. This has proved to be exactly the opposite (Kruchten, 2003) as these projects, by their nature, have to be well planned. The nature of the iterative process requires the iterations to be performed in a controlled manner and the responsibilities of the relevant players have to be well defined. Also testing is more concise as small simple modules are tested as and when they are developed rather than a large system, which needs to be tackled as a whole. The iterations in the project lifecycle have been illustrated in figure 2.3. In the figure 2.3 the four phases are run across the top of the image. Down the left side are the various disciplines that are undertaken during the life time of the project. For example for "Requirements" most of the effort is applies in the inception phase of the project. However, due to the iterative nature of the process, see the last block in the image, the requirements may change to a lesser degree during the lifetime of the project. As the project nears the end, the lifetime of this discipline reduces to zero.

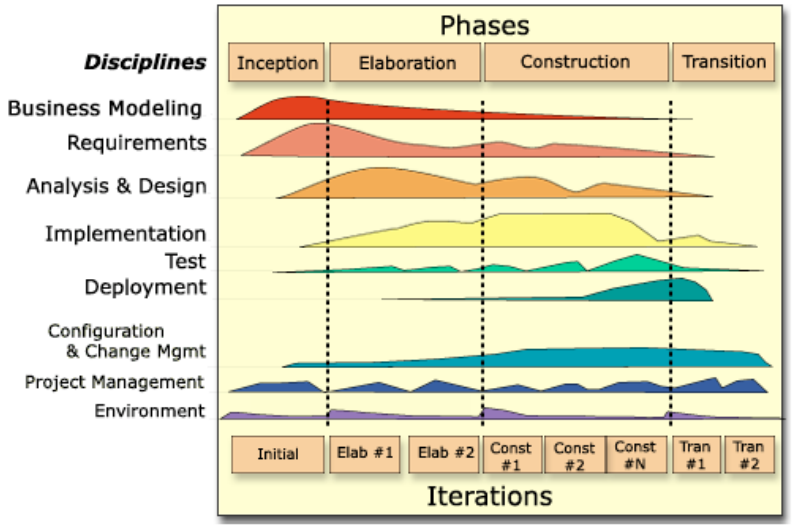


Figure 2.3: The Rational Unified Process (after Kruchten, 2001)

According to Kruchten (2003), The Rational Unified Process covers the entire life-cycle of the project and brings a wealth of knowledge of the current best methods available to users and as more methods become available so they are included into the process. It is comprised of many modern techniques and processes not least of these is the iterative process. The process is not static but changes as new developments arise, which means that the products will continuously be at the cutting edge of their field. The very nature of the RUP means that it is highly adaptable and can fulfill a number of requirements for very different scenarios. Companies that make use of this process range from Ericsson in Telecommunications; British Aerospace in transportation; Xerox, Volvo and Intel in Manufacturing; Visa in Finances and Ernst and Young in the field of System Integrators.

2.4 A management approach to Software Engineering

Bennetts et al. (2000) quoted Pressman (1987) raised the questions that "for the past decade managers and many technical practitioners have asked the following questions:

- Why does it take so long to get programs finished?
- Why are costs so high?

- Why can't we find all the errors before we give the software to our customers?
- Why do we have difficulty in measuring progress as software is being developed?"

They continued and noted that despite these questions being well known the same issues are still occurring. This indicates that although there are many Software Engineering approaches known, it is not the case that one method addresses all problems. Hull et al. (2002, pp 11) reviewed three approaches to solving these problems that have emerged from best practices and concluded that, "In many ways the results show that there is little to choose between them." Other commentators like Bennetts et al. (2000) and Rose (2002) favoured the use of SSM in information systems development, the later developing yet another "systems development concept." Rose (2002) went on to conclude that his model "views systems development primarily as a social and managerial task, rather than a technical one."

The approach taken in this study was to use a blend of SSM and the RUP methodology. This combination had the advantage of satisfying the critea that systems developmentis primarily a managerial task (Rose, 2002) and that SSM is a good metaphor for the development process Rose (2002), Bennetts et al. (2000), Lars Mathiassen and Stage (1991), Checkland and Holwell (1998), Holwell (2000). And lastly, that the RUP methodology is "presented in a very management orientated format" (Hull et al., 2002, pp 11). Not to mention, that since it is iterative and is therefore more suitable to the action research model as demonstrated in figure 1.1. The relationship between the chosen methodology and the underlying philosophical framework are outlined in the next chapter.

Chapter 3

Philosophical and Research Framework

This chapter provides the philosophical and research framework upon which this study was based. It begins by categorizing the elements required to understand messy problems within the environment. And continues to describe the research framework upon which, the methodology of SSM, Chapter 4 was built.

In this thesis the area of social research was considered. According to Cohen, Manion, and Morrison (2000) the search for knowledge regarding the environment and the way it is understood, can be broken down into three broad categories: experience, reasoning and research. Naturally, as an academic contribution the process requires rigour as well. Examining each of these areas and the way that knowledge is acquired means that the understanding of complex problems in the environment can be better understood.

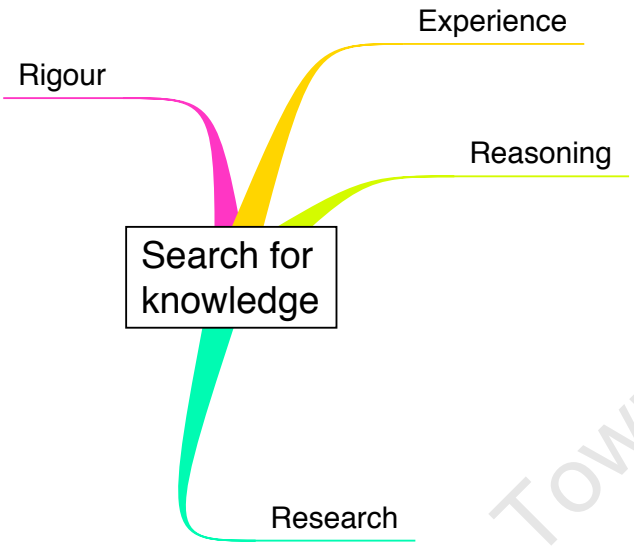


Figure 3.1: The Search for Knowledge. (After Cohen et al. 2000)

Firstly the role of experience cannot be neglected. As observers bring to bear their experiences of the environment to the data the understanding is influenced by the participants not only as observers but also as participants in the research process. However in a complex environment one needs to carefully evaluate the method in which the experience is used, to bring understanding to the research. Researchers need to carefully construct their hypotheses and systematically evaluate them. In an effort to do this scientists seek to control their environment rigorously so as to have as few variables as possible and therefore be able to explain and understand the processes at work. This often proves difficult in social research as a large portion of the variables are beyond the scientists control, and the experiment cannot be placed within a controlled environment.

The second category used to try and understand the environment is that of reasoning. According to the Peircean logic system there are three methods of reasoning. These are deductive, inductive and abductive (Yu, 1994) .

Deductive reasoning is based on Aristotle's syllogism, which means that in this form of logic a logical conclusion is drawn from the premises known. The result is true given that the premises are also true. This is best illustrated with a simple example:

All A's are B's

C is B

Therefore, C is A

Logically it follows from this that using this method one cannot gain new knowledge, because the conclusion, the "therefore" is just a restatement of what preceded it.

Inductive reasoning was introduced by Francis Bacon (Yu, 1994). Unlike both deduction and abduction which treat both propositions and assertions as equals, Peircean induction leads support to assertions. Induction is based on the notion that the premises of the argument point to the conclusion but do not directly prove it. This is illustrated below:

All A's I have seen (A_1, A_2, A_3 ... A_{100}) are Bs

C is B

Therefore all C's are A.

Although the first premise is not universal in claim, it is inferred inductively that the conclusion is true for all Cs. The result of induction is that by studying enough cases, a hypothesis, will lead to a generalization (Mouly, 1978).

Abductive reasoning in contrast has been described as not symbolic logic, but critical thinking (Yu, 1994). Abduction is the process in Peircean logic that examines a phenomenon and then suggests the most appropriate hypothesis (Peirce, 1878) This idea is illustrated below:

We observed a phenomenon A;

This can be explained by hypotheses X, Y and Z;

Z is able to explain A;

Therefore, there is reason to explore Z.

Popper's idea of falsification (Popper, 1963) to falsify each option has not been attempted here. Rather it was decided, pragmatically, to focus this thesis on understanding the phenomenon observed.

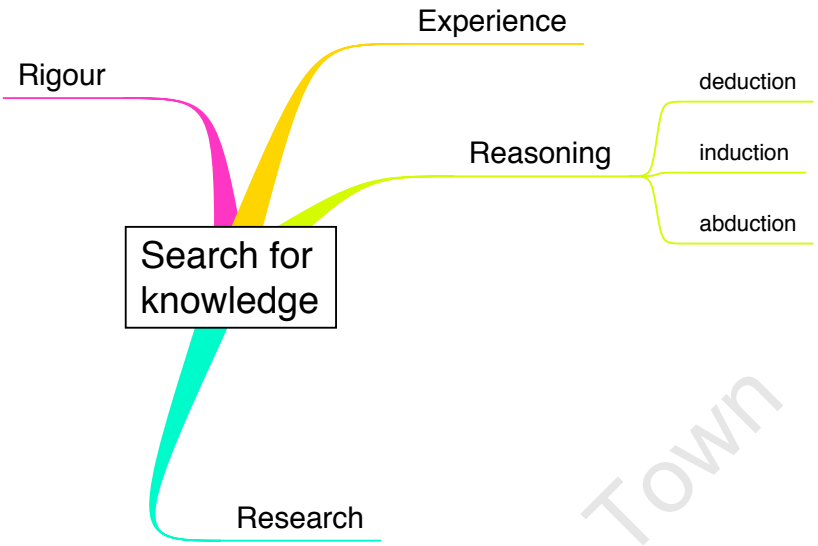


Figure 3.2: The Search for Knowledge - Reasoning

When the three methods of reasoning are examined, from a Peircean perspective (shown in figure 3.2), one can conclude that both deduction and induction each have advantages and disadvantages. From Peirce's perspective a reasoner should use all three methods of reasoning to achieve a comprehensive inquiry. Yu (1994, pp 24) summarised these methods as follows:

> "Abduction and deduction are the conceptual understanding of a phenomena, and induction is the quantitative verification. At the stage of abduction, the goal is to explore the data, find out a pattern, and suggest a plausible hypothesis with the use of proper categories; deduction is to build a logical and testable hypothesis based upon other plausible premises; and induction is the approximation towards the truth in order to fix our beliefs for further inquiry. In short, abduction creates, deduction explicates, and induction verifies."

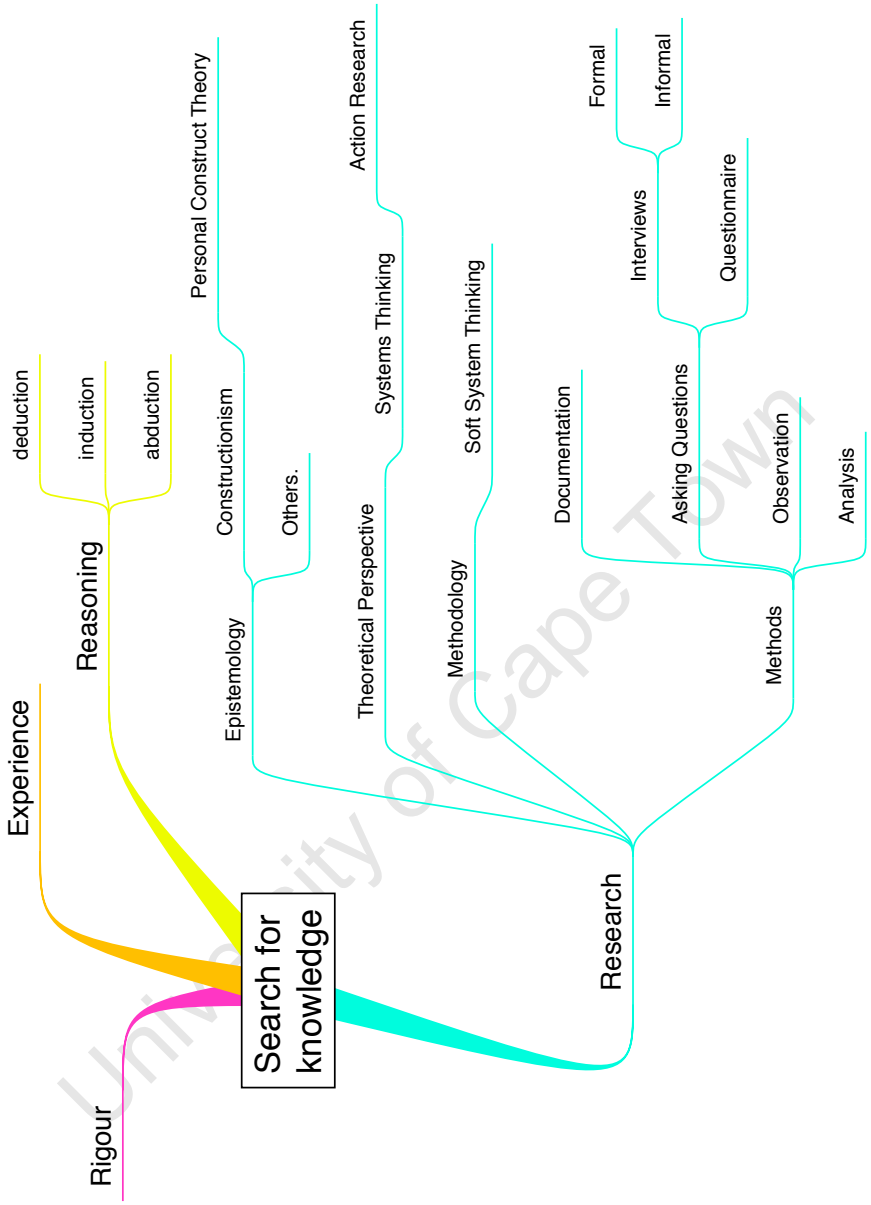


Figure 3.3: The Search for Knowledge - Research

The third category used to understand the environment is via research, figure 3.3. According to Crotty (1998) research is based on two broad proposals. Firstly when doing research which methodologies and methods will be used must be considered and secondly the reason for selecting them must be justified. Examining these

components closely brings to light a few more questions that require answering. These are epistemological questions. When research is done, which results and what knowledge will be achieved. How will the readers treat this knowledge in this research and what can they know about the characteristics of this knowledge. Thus in establishing a research framework Crotty (1998, pp 2) suggested that four questions need to be addressed.:

- "What *methods* do we propose to use?
- What *methodology* governs the choice and use of methods?
- What *theoretical* perspective lies behind the methodology in question?
- What *epistemology* informs this theoretical perspective?"

As the answers to these questions establish the research framework they need to be examined in more detail. The term "methods" refers to the techniques and procedures followed to obtain and process the data related to the research question. The methodology refers to the overall strategy that underpins the methods that have been chosen to use. The methodology is the link between the method and the results of the research. Underlying the methodology is the theoretical perspective in the philosophical framework that established the logic and relevant criteria of the methodology. At the base of the stack is the epistemological understanding. The philosophical understanding of what can be known theoretically and from the theoretical perspective what knowledge can be gained from the chosen methodology. This is graphically illustrated in figure 3.4.

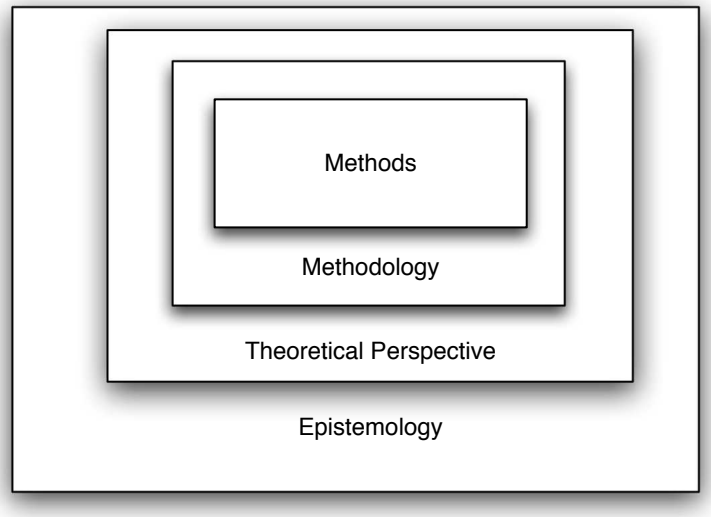


Figure 3.4: Basic elements of the research process (after Crotty, 1998)

Each of the components in figure 3.4 will be examined and the impact they have on the research framework will be discussed in the following sections of this chapter, not to mention in the chapters that follow.

3.1 Epistemological Perspective

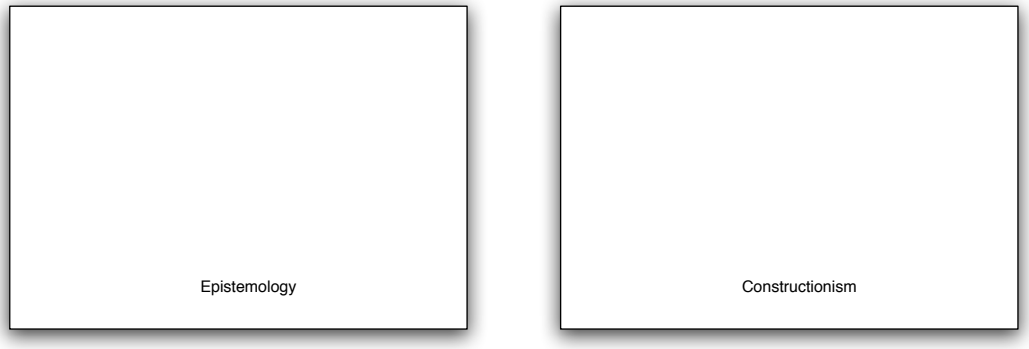


Figure 3.5: Epistemology

Epistemology seeks to answer the question of "How do we know what we know?". It deals with the nature of knowledge and provides the philosophical grounding for deciding what kinds of knowledge are possible (Crotty, 1998). The epistemological perspective chosen for this research was that of Constructivism.

Constructivism stands between objectivism and subjectivism philosophically. Crotty (1998, pp. 42) defined constructivism as "a view that all knowledge, and therefore all meaningful reality, is contingent upon human practices, being constructed in and out of interaction between human beings and their world, and developed and transmitted within an essentially social context." From this definition it is clear that our knowledge is not discovered but constructed. As such, a better definition of what constructionism is, is that "meanings are constructed by human beings as they engage with the world they are interpreting" (Crotty, 1998, pp 43). From this definition meaning or the search for knowledge, cannot be objective - because it involves human constructions and in the same way is not wholly subjective because it involved interaction with the environment. Because of its intrinsic human interpretive nature, constructionism affirms that there is no true or right interpretation, but, rather more pragmatically, that there are just useful perspectives. Utility is the mark of constructivism usefulness, rather than the objectivism "True" of "False" interpretations. Crotty (1998, pp. 58) distinguished between constructivism and constructionism, he preferred to reserve the former as "epistemological considerations focusing exclusively on the meaning-making activity of individual mind" and the latter for "the collective generation [and transmission] of meaning." In this thesis we will use constructivism to mean the latter as this is the meaning commonly found in the literature cited.

Zuber-Skerritt (1995) used the concept that each human being is a "personal scientist", introduced by George Kelly in his Personal Construct Theory. As such, each person is involved in observation, action and control of their environment. The way in which this is done, Kelly suggested is to construct a representative model of the world which guides us in the behaviour and action in the world (Zuber-Skerritt, 1995). Thus, as a personal scientist, one can use constructivism for the systems in which we act. Midgley (2001) concluded that contrary to those who argue that philosophy restricts practice, it does in fact establish the connection between methodology and methods.

3.2 Theoretical Perspective

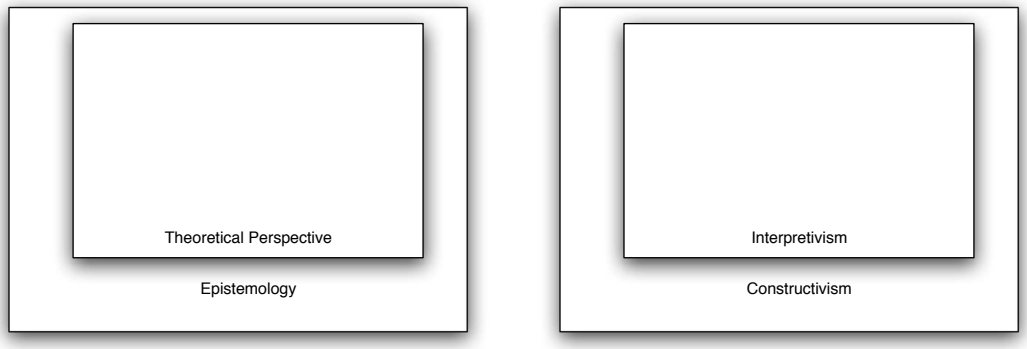


Figure 3.6: Theoretical Perspective

In order to understand our world, Descartes proposed a reductionist model with which to interact with the environment. In contrast systems thinking provides an inclusive, holistic model believing in the Gestalt theory that the whole is more than the sum of its parts. Systems thinking recognizes the fact that unlike the natural sciences where it is possible to test hypotheses under restrictive laboratory conditions with real world issues it proves extremely difficult to repeat this type of process. Secondly, real world problems are not repeatable in a laboratory environment and need to be examined in-situ to gain the most knowledge (Jackson, 2000). System thinking is employed to gain knowledge about the relationships and interactions of the various components of the system in focus. Broadly viewed a system is defined as group of components or parts that interact together within a particular time frame. These systems are generally embedded in larger systems, known as supra-systems, and within the systems the components may themselves form a system, known as a sub-system. Jackson (2000) quoting Ackoff said that the “machine age” of the industrial revolution was appropriate for Descarte’s reductionism but systems thinking is the appropriate response for the birth of the systems age. Systems thinking is a broad category used to distinguish this research study from reductionist perspectives. In this study, interpretivism was used as the theoretical perspective as it emerges naturally from constructionism, the chosen epistemology. Walsham (1995) investigated the history of interpretivism in IS research and introduces the concept of interpretivism. "Interpretive methods of research adopt

the position that our knowledge of reality is a social construction by human actors. In this view, value-free data cannot be obtained, since the enquirer uses his or her preconceptions in order to guide the process of enquiry, and furthermore the researcher interacts with the human subjects of the enquiry, changing the perceptions of both parties. Interpretivism contrasts with positivism, where it is assumed that the "objective" data collected by the researcher can be used to test prior hypotheses or theories" Walsham (1995, pp 376). Interpretivism forms the basis of the methodology which follows.

3.3 Methodology

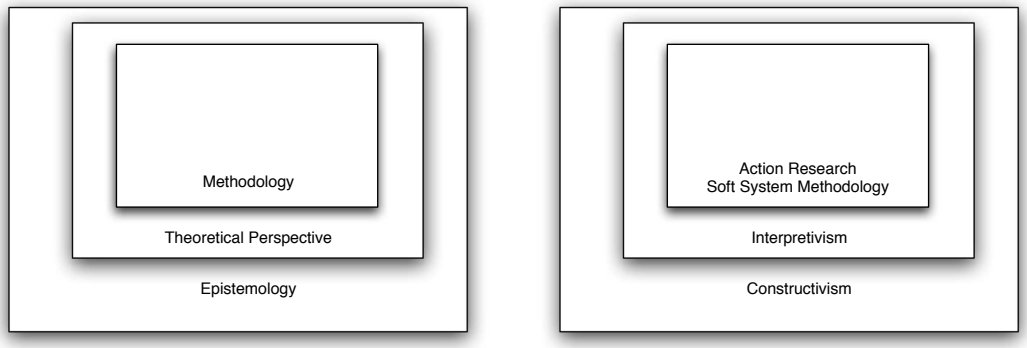


Figure 3.7: Methodology

According to Dick (1993):

> "There are many ways to do action research. It is a research paradigm which subsumes a variety of research approaches. Within the paradigm there are several established methodologies. Some examples are Patton's (1990) approach to evaluation, Checkland's (1981) soft systems analysis, Argyris' (1985) action science, and Kemmis' critical action research (Carr and Kemmis, 1986). Each of these methodologies draws on a number of methods for information collection and interpretation, for example interviewing and content analysis."

Action research was started by Kurt Lewin a German psychologist in the 1940's. Interestingly he advocated Gestalt psychology. In contrast to the accepted method

of dissecting problems he recognized that social problems are inherently complex and could be best studied *in-situ*, in action. Essentially action research consists of the following components: planning, acting, observing and reflection. These actions are taken in a cyclical manner and action research involves both action and research for knowledge at that same time (Dick, 1999). As these cycles progress from acting to reflecting the methods used are developed. This process is said to be emergent. Action research is an established research paradigm that has "proven its utility, with growing recognition of its breadth as a field of research practice and its depth as a discourse and of theoretical insight" (Altricter et al., 2002, pp 125). "Action research simultaneously assists in practical problem-solving and expands scientific knowledge, as well as enhances the competencies of the respective actors, being performed collaboratively in an immediate situation using data feedback in a cyclical process aiming at an increased understanding of a given social situation, primarily applicable for the understanding of change processes within a mutually acceptable ethical framework" (Hull and Lennung, 1980, pp 247). The particular methodology used in this research project is Peter Checkland's Soft Systems Methodology and is described in detail in Chapter 4. "SSM embodies a philosophy of organizational intervention that sees different individuals and groups as constructing interpretations of the world, the interpretations having no absolute or universal status. The purpose of the intervention is to reconcile these views sufficiently to achieve organized action" Walsham (1995, pp 379).

3.4 Methods

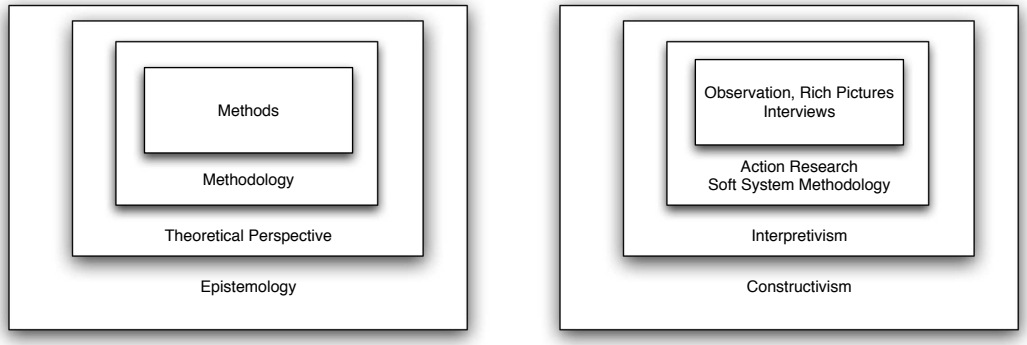


Figure 3.8: Methods

The methods used to collect data for this thesis were those that are traditionally used in social science studies. These include both formal and informal interviews, data collection from documented sources such as formal meetings, informal interviews, project review meetings, and email as well as the drafting of rich pictures. The methods are discussed further in the chapter regarding the methodology for this thesis, Chapter 4.

Chapter 4

Soft Systems Methodology

In this chapter the historical background and process of SSM were used to derive a working definition of SSM as an Action Research Paradigm and then used to provide solutions for messy soft system problems within a managerial environment. The Soft System Methodology (Checkland, 1981, 1999) was chosen for this project because of its suitability to help understand messy IT problems. SSM was developed in the 1970s and 1980's by Peter Checkland. In response to the failing of the hard systems engineering approach of the 1950's and 60's, Checkland developed SSM to better understand and tackle problems where the hard engineering methodology had failed, specifically technological projects with ill-defined and changing parameters, such as "messy" problems. In order to understand the need for SSM the failure of hard systems engineering (according to Checkland) and the "emergence of SSM as a response to that breakdown" (Checkland, 2003) needed to be addressed. In this chapter it has been demonstrated that SSM is a methodology that is well suited to be used to tackle ill-defined problems in such a way as to bring about systemic change and solve messy problems.

4.1 Outline of Soft Systems Methodology (SSM)

In hard systems the focus is on *how* to solve the problem. The systems are well defined, and the question the researcher asks is, "*How* must I do something in order to achieve the objective, which is defined up-front?" Checkland (2003) illustrated this with the example of this type of systems engineering approach by using President Kennedy's Rice University speech in 1962. In his speech Kennedy defined the

objective as: "before the end of the decade, to land a man on the Moon and return him to Earth." (Kennedy, 1962). The objective of the problem was well defined - land a man on the moon and return him to earth and left just the *"How* do we do it?" to be implemented.

In contrast the Soft Systems Methodology focus is not based on a paradigm of optimization of the system, but rather on a paradigm of learning more about the system (Jackson, 2000). Soft Systems Methodology (Checkland, 1981, 1999) addressed problems were the hard systems approach fails. In the case where the problem statements are "What are the objectives?" and "What are we trying to achieve?" (Checkland, 2003) a hard systems approach fails.

To address these messy types of problems SSM emerged in the late 1970s. The question that SSM seeks to answer *is "What* should be done?", rather than *"How* should we do it?". It is precisely in seeking "What should be done?" that more was learnt about the systems involved and through that understanding, behaviours could be modified to achieve the desired outcome that emerged as understanding deepened.

In seeking to answer this question, multiple viewpoints are considered as part of the decision making and intervention process (Flood and Jackson, 1991). As a result of this participative method of problem solving "Checkland rightly argued, therefore, that the social theory implicit in his methodology is interpretive rather than functionalist, and that its underlying philosophical base is in phenomenology rather than positivism. In soft systems methodology, systems are seen as mental constructs of observers in the world" (Jackson, 2000, pp 248-249).

According to Flood (2000) there are a number of important strands that are particular to SSM. "These are the realizaton of a unique brand of action research, and from this the crystallization of an interpretive-based systemic theory, as well as the establishment of principles from action in ill-structured problem contexts that he labeled soft systems methodology" (Flood, 2000, pp. 724). Interactions between SSM and action research, as well as SSM and interpretism and the suitability of SSM to messy IT problems were considered.

Action Research concepts where used by Checkland in the development of soft systems methodology (Baskerville, 1999). These action research principles included those of "subjectivity, learning, multiple views" (Holwell, 2000, pp 778) "Checkland applied such action research principles to a program of research for the management sciences and in particular systemic thinking. The program generated a

framework of ideas that might be known as an interpretive-based systemic theory" (Flood, 2000, pp 725). The Action Research Paradigm is the chosen paradigm for this thesis, because of its practical approach to problem solving and ability to involve multiple perspectives in its problem solving approach. This approach is interpretivistic in nature. Holwell goes further and "has argued that an interpretive foundation is a necessary (and a defining) characteristic of SSM" (Holwell, 2000, pp 775). Flood (2000, pp 726) argued that a "new angle on framework of thought, methodology and action area was worked out through action research. The result is congruent with an interpretive-based systemic theory" known as SSM. In her review of SSM Holwell (2000, pp 778) said "Checkland's work is recognized for adding interpretive thinking to the field of systems, problem-solving, and information systems (IS) : so much so, that his argument and language have become part of the general discourse." It has been shown in section 1.3 that SSM has been used in dealing with IT problems, particularly the class of problems known as messy problems. Flood (2000, pp 726)stated that "Emerging from this reconceptualisation were principles for action in ill-structured problem contexts. The principles became known as soft systems methodology."

4.2 Underlying principles of Soft Systems Methodology

Flood and Jackson (1991) identified four main principles that underlie the implementation of SSM that one needed to be cognizant of when employing the methodology. These emerge out of the development of SSM and concern learning, culture, participation and the differing modes of thinking.

Since SSM addresses problems and the management there of, it naturally takes a view on what management *is* and what a manager *does*. Management operates in the world which by its very nature is in a state of constant flux as various systems interact with one another. Checkland (2003) defined management as a "means of reacting to that flux: perceiving and evaluating (parts of) it, deciding upon action, and taking action which itself becomes part of the on-going events/ideas flux, leading to new perceptions and evaluations and further actions."

The purpose of SSM is that of learning. As a learning system its objective is to learn with the objective of taking action from what I learnt and, as the systems changes to implement new changes within a continuous cycle. This is unlike the hard systems approach which is single goal orientated moving from means to end in the most

optimized way. The learning component is that action which the manager does before deciding what corrective action to take within the system he is managing.

The second aspect addressed by SSM is that of cultural fit. As SSM is highly participatory in nature any action taken within the system ought to reflect this and have a high cultural fit as it is implemented within the "real-world".

As shown, SSM relies on the principle of participation to obtain multiple perspectives on the problem space. It can be argued that if there is little or no participation, that true SSM cannot take place as the modifications in the "real-world" will not be successfully implemented.

The last process that underlies SSM, different modes of thinking, that has been alluded to is that of functioning in two realities, one of the "real-world" and the other in the "system thinking space" The thinking behind this is that adjustments are made in the idealized systems space and implemented in the "real-world" where the effects are fed back into the systems spaces and can then be further refined.

4.3 Description of Soft Systems Methodology

Soft Systems Methodology has evolved over time since first published by Checkland (1981). In later publications Checkland modified the methodology and even specifically applied it to Information Systems (Checkland, 1983, Checkland and Scholes, 1999, Checkland and Holwell, 1998, Checkland, 1999). Checkland and Scholes (1999) contained a comprehensive discussion on the development of SSM. In this section the classic seven stage methodology that underpins all these variations of the methodology have been outlined.

Stage 1: Collecting the Data

In this stage data is collected using a number of different methods. Data can be gathered from a variety of sources such as observation, formal sampling procedures, meeting minutes, email, formal and informal interviews etc. Most importantly when collecting data about the "mess" one must not impose a direction on the data. Questions should be open ended and are commonly in the form of "What?" , "Why?" and "Describe?'

The purpose of this stage is to gain enough data about the problem in a unstructured way. Once completed the manager can move on to the next stage.

Stage 2: The Rich Picture

In this stage, a graphical representation of the data in the form of a rich picture is constructed. A rich picture is a snap shot of the data captured by the researcher and presented in the form of a cartoon like picture. This image contains a representation of the structure and the attempts to document the relationships between the various components, this is known as the climate of the situation.

Stage 3: The Root Definitions

From the rich picture root definitions (RD) are defined. These are idealized statements which concern the system and to ensure that they are well formulated they follow the mnemonic CATWOE, that was determined by Checkland (1981) and are explained in Table 4.1.

C	Customers: who would be victims/ beneficiaries of the purposeful activity
A	Actors: those who do the activities
T	Transformation Process: the purposeful activity which transforms the input to the output
W	*Weltanschauug*: the view of the world that makes the definition meaningful.
O	Owner: who can stop the activity
E	Environmental constraints: this constraints in its environment that this system takes as a given.

Table 4.1: CATWOE from Flood and Jackson (1991)

The core of this process is the **T**, the transformation, which ”represents the purposeful activity to be modeled, expressed as a Transformation process” (Bergvall-Kareborn et al., 2004, pp 60). This leads to the next stage which is developing the conceptual models.

Stage 4: Development of Conceptual Models.

The conceptual models are built on the principle that Checkland (2003) described as very simple and yet very sophisticated. Simple, in that they are based on something we all know - verbs, and sophisticated because there are so many of them to choose from. The models use only the RD and are constructed by using the minimum amount of verbs to describe the system outlined in the RD. Based on the

work of Miller (1956), Checkland recommended to aim for 7±2 activities. Some of these components should involve the operational activities to the system and a few should be concerned with the moderation and control of the conceptual system. Flood and Jackson (1991) pointed out that is it useful to think ahead to stage 5 when developing these models so as to save time devising uninteresting models that do not prove to be useful, necessitating re-looking at stages 2 and 3 of the system to redefine the root definitions. The result of this stage is a model that is of a system that is able to survive via interaction with its environment by adapting using the processes of communication and control. This conceptual model is illustrated in figure 4.1.

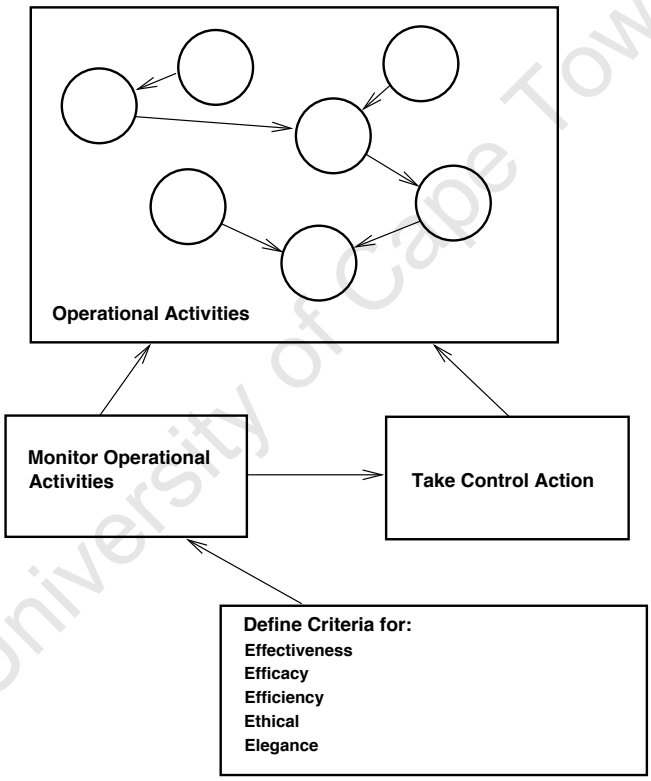


Figure 4.1: The general structure of a model of a purposeful activity system from Checkland (2003)

The criteria by which this transformation is judged is by means of Checkland's 3 E's, subsequently expanded to 5 E's Checkland (1999, 2003). The 5 E's are outlined in Table 4.2.

Efficacy	Does the means work to justify the ends?
Efficiency	Are we using the minimum and correct resources to complete the work?
Effectiveness	Does it work? Does this process **T** help attain the goals of the O's expectations?
Ethically	Is **T** a moral thing to do?
Elegance	is **T** aesthetically pleasing?

Table 4.2: The five E's of the Monitor Operation Control System, from Maxson (2005)

Stage 5: Comparison of the Models to the "Real-World"

Once the model is defined, the next stage is to compare it with the "real-world". The purpose of the stage is to generate debate. This debate centers around what differences and similarities exist in the conceptual models of stage 4 and the "real-world". Checkland (1981) made a number of suggestions as to how we can best make use of the stage.

Firstly, on the modeling side, a number of different models can be generated and compared. Secondly, the differences for the conceptual model and the "real-world" can be formally listed and the differences annotated and an explanation sought. A third approach would be to work with the conceptual model on paper and do a dry run as to what the models outcome will lead to in the future. This outlook can then be used to feed back information into the debate.

The outcome of this stage is an agenda for debate with the actors of the system, identified before in stage 3. This agenda highlights topics concerning, for instance: the models mismatches and omissions, but avoids discussing real-world prescriptive solutions (Waring, 1996).

Stage 6: Defining What Actions will be Taken

Inevitably discussion about the models will give rise to a list of possible changes that need to be considered as actions to be taken. Defining these actions is the purpose of this stage. In order to finalize the action to be taken changes need to be evaluated if they are both desirable and feasible. This feasibility includes looking at the cultural feasibility of the actions as well.

Stage 7: Implementation, Taking Actions

In this last stage the actors now have an agreed list of changes that need to be implemented. SSM can be used at this stage again to implement these changes.

By examining the SSM process and the development of the SSM process historically it can concluded that SSM provides a good and appropriate methodology to use for the type of messy problem that was considered for this thesis. The data collection method is integral in bringing all players to the table. The rich picture enables all players to understand the entire project and so take the appropriate role within the process. The root definitions, defined in stage 3, using the CATWOE process, ensures that all players continue to remain involved and means that the transformation process is well-defined within the system environment.

The operation activities defined by the conceptual models in stage 4 reduces the "messy problem" to a more manageable transformation based on Checkland and later Maxson (2005) 5 E's: Efficacy, Efficiency, Effectiveness, Ethically and Elegance. While stage 4 appears to remove the model from all the players in stage 5, by comparing the generated model with the real world, returns all players back into the discussion ready to contribute to stage 6 which defines the worthiness and practicalness as well as ethicalness of actions to be taken on the model when compared with the real world. In the final stage, where a list of actions is defined, the project is ready to proceed in a meaningful way, since it has had ongoing contributions from all players and ongoing constraint by comparing the model environment to the real world.

Chapter 5

Soft Systems Methodology in Practise

In this chapter the Action Research Process has been demonstrated as a real-world example, that of the deployment of a mobile communication platform described in Section 1.3. The company in which this Action Research study took place was formed approximately at the time of the bursting of the Dot-Com bubble. This high technology mobile communication company was launched in 2000, largely funded by venture capitalists, its aim was to develop a telecommunications grade mobile communications platform. It took around 18 months before the company started producing a product that was ready to be used commercially. In the interim, a pilot system was developed in parallel to show proof of concept, and placate the investors. Although developed using rapid prototyping tools, they were suitable for the purpose designed as the mobile space in which they operated was a new technological space and underutilized. This system, known as the "junior system" was developed and operated by the developers. One of the issues going forward that hampered the companies focus was the reluctance of the developers of the "junior system" to fully migrate their applications from the junior to the senior system.

The company consisted of around twenty five staff, of which ten where directors. The balance of the company consisted of developers, a marketing department and the administration department. Technical expertise not met within the company was outsourced to a variety of individuals. My first involvement with this company was as an external consultant. I was consulted as an operating systems and inte-

gration specialist and I was involved in the company for approximately eighteen months before being employed on a full time basis as the operations manager and internal consultant for research and development. This research was carried out over the time period from 2003 and 2004 in my capacity as a company employee.

Previously to joining the company full time an installation occurred which was the companies first international installation and occurred in Tanzania (November 2001) for a large emerging mobile communications company. The work was scheduled to take a week and was to be done on-site in Dar Es Salaam, Tanzania. The team consisted of the system architect, a developer, a marketing director, the company CEO and myself as consultant operations and integrations consultant. The installation overran both in budget and time. The installation took over two weeks and was plagued with numerous bugs, integration and stability issues. It required a team of developers patching code in the office back in Cape Town each day and resubmitting to get the platform up and running. What made this all the more remarkable was that this platform was installed in an environment in which the company had the most control of all the external variables of all their subsequent installations. The system was hosted in the mobile communications companies data center, in the same location as the Short Message Service Center (SMSC), with direct connections to all the relevant platforms that were needed to configure the mobile communication platform.

The first iteration of the Action Research Process corresponds to the second installation, which was again in Tanzania ten months later, in September 2002, this was also the month I started full time employment at this company. Having being involved in the first installation and witnessed first hand the issues involved a much smoother installation was expected. Again this was not the case. The second installation was expanding the services offered on the platform as well as a platform software upgrade. Once again issues were with the deployment and configuration of the platform. Considering that these platforms were installed in known environment with well understood constraints these issues should not have occurred.

The second iteration in the Action Research Process corresponds to the third installation, which followed a year after the Tanzania installation and was in the United Kingdom, it was a hybrid system developed specifically for the task. The installation took place in September 2003. It had been running in a hosted environment in South Africa, but as the platform was mission critical to the outside organization, they felt that they needed more control over the system and required a dedicated

platform on-site. This installation proceeded a lot more smoothly than the previous installations, but was also a much simpler application.

The third iteration in the Action Research Process corresponds to the forth installation, which followed shortly afterwards in November 2003, in UK. During the intervening months the home development environment was perfected with the use of mobile phones from the host country. The idea behind this was to replicate, as much as possible, the future hosting environment of the platform.

As Operations Manager my task was to keep the operational cost within acceptable limits and to reduce the operational costs to the company both during installations and subsequently, through the maintenance and operation of these platforms. In order to discover the reasons for the highly variable results of installation and to answer the CEO's request of "not strapping a techie to each platform" I used the Soft Systems Methodology to discover what had happened during these and subsequent installations.

5.1 First Iteration of the Action Research Process, Tanzania (2002).

In this section the research question determined in Chapter 1 has been applied to the SSM developed in Chapter 4.

Stage 1: Collecting the Data.

In order to proceed I collected the data needed. This data came from a number of sources. Firstly the company already had established a weekly Operations and Developers meeting. This was a source of report back from the operations and development teams, and although informal minutes were kept, the source of data used in this study were from my personal notes. Examples of these are attached in Appendix A as Document One and Two. Secondly the next source of the initial data collection was from *adhoc* meetings between the operations team, the lead developers and systems architect . Although infrequent, these meetings provided more specific data with regards to the implementation of the developing system and were also the forum in which the feedback and discussion of the next actions took place, such as Document Three and Four in the Appendix. The third source of data

was in the form of informal interviews with rest of the developers and operations staff. Data from these meetings were captured in a journal and formed the basis of the rich picture that emerged in the next stage. Formal feedback meetings were held by the company's Chief Technical Officer (CTO) to evaluate the installation procedure and formulate new plans as to what needed to be done to move the process forward for example Document five in the Appendix. These meetings were held at a higher - director - level and while they prescribed what needed to be undertaken details were left to individual teams.

Stage 2: The Rich Picture

The second step that was taken was to construct a rich picture, that is to capture a snapshot of the processes and systems involved. Data obtained for this rich picture came from the involvement with the initial installations, emails, informal interviews undertaken with both management of the company and with the developers and formal meetings after the Tanzanian trip to debrief on what had happened and what went wrong. The operational and development teams had a standing Monday morning 9.00am meeting where most of the issues that affected all the developers and operations were discussed. Meetings between the Systems Architect, Operational Manager and the Senior Developers were also held on an *adhoc* basis. The aim of these meetings was to discuss high level changes to the system. This was the forum were changes to the system were initiated. Drawing on these as sources for data, a rich picture of the problem situation was drawn up. The rich picture, figure 5.1, is a snap shot of the problem space in November 2002, and seeks to capture as fully as possible the problem space. Table 5.1 provides a key to the key elements of the Rich Picture.

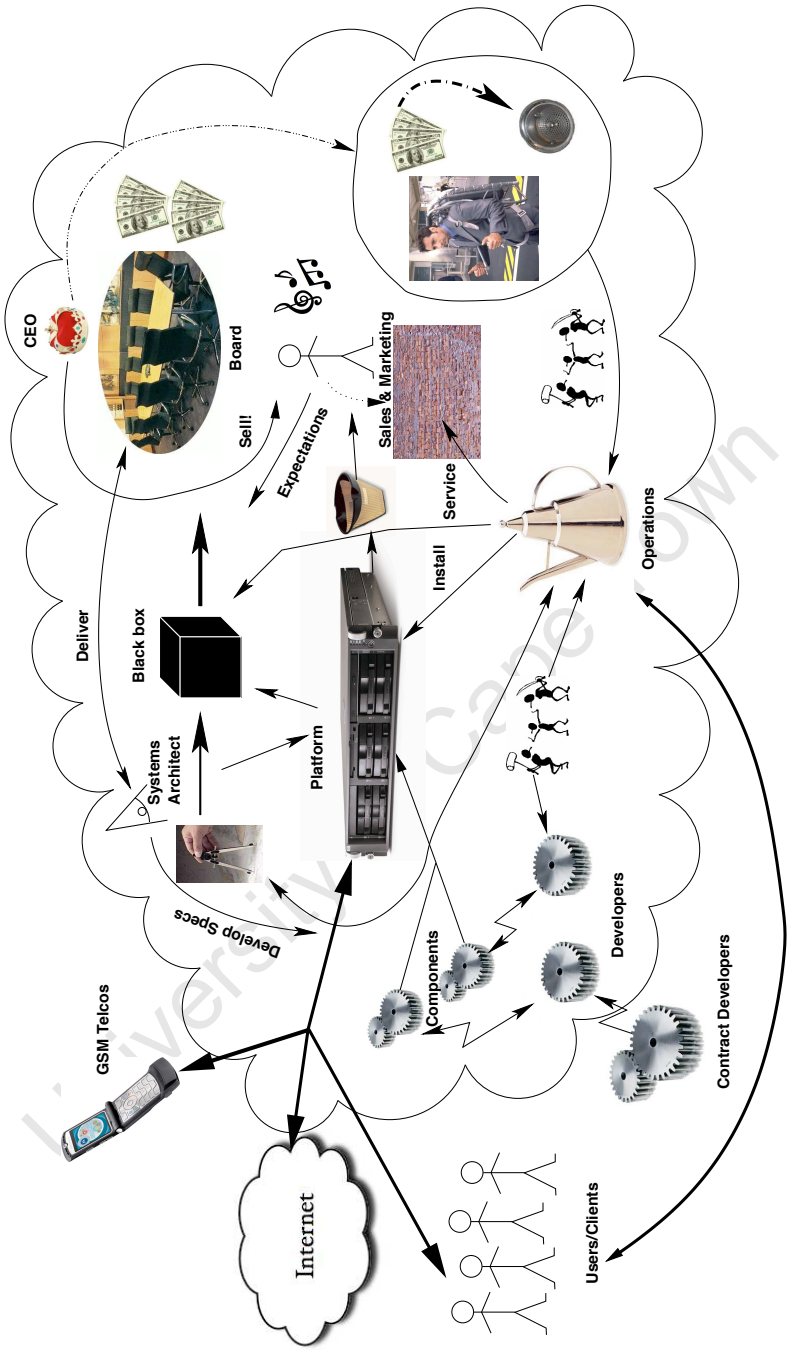


Figure 5.1: Rich Picture as of November 2002

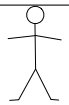
Graphics	Key
	The Board
	The Chief Executive Officer
	Developer
	Chief Technical Officer, the Systems Architect
	Marketing team
	clients
	Operations team

Table 5.1: Key element's of the rich picture

The Rich Picture shows the relationships between various groups within the system. They are broadly categorized as the Board of Directors and CEO who were the financiers of the operation. The System's Architect or Chief Technical Officer, represented as an "eye", was the person who has oversight of the development specifications and implementation of the platform. The Lead developers and developers (gears) were responsible for the actual implementation systems architect specification of the code. The marketing team, seen as the singer in the rich picture, was responsible for marketing the product, the platform and the services it provided to another group, the clients. Although clients were seen as external, there were also internal clients. These clients were serviced by the operations team (the oil can) who were responsible for communicating information back to the developers, marketing and ensuring the smooth running of the system.

In this snapshot in time (November 2002) the following relationships existed in the company. The CEO and the Board had asked the Systems Architect to develop the platform, simultaneously tasking sales and marketing to start marketing the product to be. The systems architect had already gathered the requirements for the various stake-holders and designed the system specifications. These were the basis for the marketing and sales departments to go and sell products and services. The Systems Architect had communicated the requirements to these developers, who were in the

process of developing code and debugging the existing products. These developers were both with the company and outsourced, with the balance in the company. The components that the developers produced were integrated by the lead developers into a coherent platform that was released to the clients in production. Initially operational support was done by the developers, but as the product matured, this task was taken over by the newly formed Operations Department.

The Operations Department was responsible for the installation, configuration and maintenance of the software and hardware platforms. They were also the customer interface to the company needing to deal with clients, the users of the platform, as well as off site network providers, such as that at each of the telecommunications connectivity providers. As the clients reported errors and bugs, this was fed back to the developers who needed to schedule time to fix these bugs. This had the potential to cause conflict between these two groups as their core focuses differed. Operations was also servicing the internal clients as well as many SMS marketing campaigns. The major internal client was the marketing department, who were using the platform in demonstrations to clients. There was a very walled mentality between these groups, with marketing making demands on operations/development that they felt were not being serviced timeiously. This led to great frustration on both sides of the "wall". Coupled to this was the marketing department filtering what the platform was actually capable of accomplishing. They were working form the original specification sheets, which had not been updated to reflect the development stages The result was they were marketing and selling features that had not yet been developed. This also caused conflict with operations who were requested to "enable" these features for the new client. This resulted in operations pressurizing the developers and placating both the clients and the marketing team.

The other requirements on operations was to streamline the installation procedure of the platform. Clients could either opt for a hosted solution or a dedicated one. The dedicated solutions were often installed on the clients premises. Due to the complexity of the system and its configuration it was taking over a week to get the platform installed. The CEO requirement was not to have an engineer strapped to the each installation as it was felt that this was money down the drain. From the CEO, marketing and clients perspective - this system should be a black box. That is it should accept the input, do its work and produce the output without the client needing any knowledge of the mechanics inside.

Another aspect of the system in focus was the Operations Management respon-

sibility to manage the platforms connections to the Short Message Peer-to-peer Protocol (SMPP) servers of the various telecommunications service providers.

Stage 3: The Root Definitions

The rich picture was drawn up from information within the company and consisted of emails, formal documentation, informal interviews and formal meetings. From the rich picture the root definitions of the system were constructed. From the rich picture a number of sub-systems were identified. With each cycle throughout the methodology the root definitions were examined to check that they were still consistent with the system in focus. The result is that the RDs were evaluated for fitness with each pass. As the platform matured some changes occurred and the system in focus was refined. The first set of root definitions were as follows:

C	Investors in the company, the board.
A	The Operations team.
T	Individual platform components developed by individual developers that are transformed into a operational communication platform by the System Architect.
W	The *weltanschauung* is that of the high technology telecommunication information technology space.
O	The owner is the Systems Architect.
E	The prevailing environment is that of the shadow of the Dot Com bubble burst and the of the high availability environment of the telecommunication industry.

This produced the root definition of the human activity system as:

> A Systems Architect owned platform, that operated within the telecommunication industry that assembled/compiled individual platform components to construct a mobile communications platform that satisfied the requirements of the investors in the company.

The *high availability* environment was defined by the market and in this case related specifically to the telecommunications sector. In this environment high availability was often described as 99.999% up-time (this translates into around five

minutes downtime per year). Telecommunications grade equipment, that was hardware and software needed to be scaleable, remotely manageable, reliable and resilient - able to cope under critical conditions, such as that of high traffic load. Other desirable features were hot-swap hardware, software that can be swapped out without shutdown to the system and redundancy to reduce a single point of failure.

Stage 4: Development of Conceptual Model

The conceptual model was derived from the root definitions. These models consisted of just the components that were logically required to have the system defined by the root definitions to function.

The generic form of the model found in Checkland (2003) (see figure 4.1) was used as the basis for the model in this section and consisted of the components required to explain the RD.

Using CATWOE, as defined in the previous chapter, as the basis for the contents of the conceptual model developed, the model developed in the first iteration of the methodology consists of the four components as shown in Figure 4.1. The Operational Activities which was the "**T**" in CATWOE were specified in more detail in the model. These Operational Activities needed to be monitored (managed) and depending on the outcome, action may have needed to be taken to control the system. This control was exercised on the Operational Activities.

The Operational Action part of the model consisted of the Transformation from CATWOE. The transformation that took place in this system was one of selecting components, developed either by individuals or teams of developers and incorporating them to form a product that best fitted the requirements and specifications of the clients. That is to say the way in which these components were chosen depended chiefly on the customers' requirements. These requirements were outputs from the marketing and sales teams as they met the clients needs for a specific customized solution. Technically these requirements were defined in a suitable way by the system architect and then the platform as a whole was installed and configured by the operations team. The operational team also had the responsibility of performing the installation, customizing further to meet the clients changing need and potentially also involved in the day to day operation of the system. The operations team also passed feedback of any changes to the system or requirements to

the system architect and developers. Figure 4.1 was applied to the first iteration in figure, and defined the purposeful activity system for this particular study.

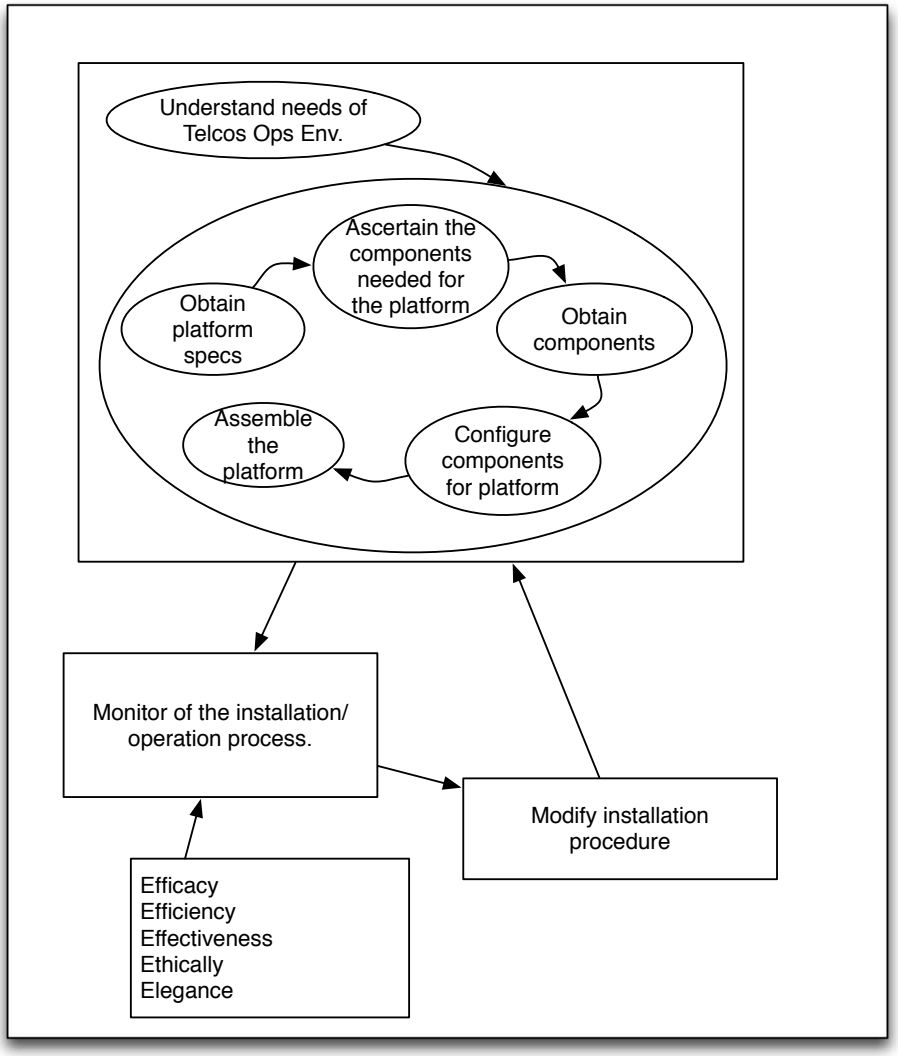


Figure 5.2: Model One, derived from Root Definitions (November 2002)

Stage 5: Comparison of the models to the Real-World

The aim at this stage was to understand the problem and the gaps in the knowledge better, so that potential improvements both to the model and the rich picture could be identified. The purpose of this was to effect change in the real world. This conceptual model was evaluated with respect to the 5E's (see section 4.2) and the

steps needed to make these changes a reality have been tabulated. In this case a comparison was made between what was actually done in the field (the real world) and what the conceptual model illustrated.

The rich picture and the model had a large number of items in common: the platform and components, requirements and how to configure and assemble them in such a way as to meet the clients requirements.

How did the methodology followed by the developers in their interpretation of the specification impact on the final functionality of the platform? How was the process managed? These questions did not seem to be considered in, nor did they feature in the model. Potentially the RD of this system could have been incorrect?

The main area missing from the model was that of the developers and their role in producing the components, the management of that process that ultimately produced the product. This raised the question - was it important to consider the developers in the process or the management of the production of the platform?

Efficacy If these components were self contained or had well defined interfaces then this model should have worked.

Efficiency If the components associated with transformation process were functioning correctly - this would be an a effective methodology to compete the installations.

Effectiveness Done correctly the owner would have a functioning system and the system would effectively meet the long term goal of the System Architect.

Ethically There were no ethical issues involved in this model.

Elegance If this model worked, it would streamline the process of the installation of the platform, and would be an elegant solution.

Stage 6: Defining Which Actions will be Taken

On paper this looked like a feasible model that had a good fit, however in reality things did not go according to plan. The main reasons isolated for this were based on the assumptions made up-front. For example it was assumed that the components were plug and play meaning that each component was wholly self contained

with well specified interfaces, and although they had well defined inputs and outputs that was not the complete story. Most components had complex configuration parameters and often depended on third party applications and services to function. Although somewhat known to the System Architect and operations team; they were not well defined and the default parameters were not suitable for the various deployment environments. No clear documentation existed as to what needed to be configured for these components to function outside of their development environment and more over these variables were not made visible to the configuration system.

In a post installation meeting held by the system architect and the operations team, the process modeled was debated and some suggestions as to a way forward were proposed. Initially it was thought that the system may be too complex to be customizable as there where a lot more variables that the platform as a whole had little or not control over. These variables were outside the direct control of the development team, such as the method in which telecommunications companies implemented the SMS specifications, the type of network protocols to connect to the required SMSC, be that over the net or via dedicated leased data lines. These factors were considered in the subsequent rounds of the methodology. For the first round it was suggested that the focus be on the development and testing of these components to ensure that they functioned as expected and had no unspecified internal and external dependencies. A change in process needed to take place in order for the deployment and operational costs to be reduced and better managed. This required a policy change from the top so that developers and development teams followed the new processes and procedure.

Stage 7: Implementation, Taking Actions

The first iteration of the SSM yielded some interesting results. In order to achieve these results a few changes needed to be made to the way in which the software was developed and tested. After extensive meetings and discussions a document was drafted outlining the way forward for the year 2003. This was included in Appendix A as Document Six. The system-in-focus was modified from just the assembly of the platform alone, and expanded to include the platforms integration into its deployment environment. Until this stage each developer was given a specification of the input types they needed to deal with as well as the outputs that there component needed to provide. What was not specified was how the component

was to be integrated into the system as a whole and where if should get its configuration details. In order for the developers to be able to test their code in a simulated client environment a private subnet was firewalled off from the main developer IP pool and was set up as closely as possible to mirror the clients environment. This caught a number of issues with configuration files and services that where being hosted on the platforms.

Armed with new found confidence in the integrity of the unit testing system and development methodology changes the team once again set off for Tanzania to expand their services and add a new platform with more features in July 2003. This proved a great success and it was thought that the installation and configuration issues of deployment had been solved. However, in September 2003 another installation at a multinational company in the UK was undertaken and the results were disappointing.

5.2 The Second Iteration, in the UK in September 2003.

Changes from the first round of SSM had been implemented into the management of the platform process, but still some issues remained. During this installation round in September 2003 the platform was deployed at two different clients in the UK. The first installation consisted of two platforms. The first platform was a service that was to be integrated to the clients existing IVR offering adding SMS capability to that service. The second platform was the installation of some of the standard products, like sms to email to sms and sms broadcasting, to their existing infrastructure. From the platform companies perspective the code development stage as a whole was nearing completion so on re-examining the methodology a new rich picture was drawn up. At this time the software engineering methodology had migrated from the traditional waterfall method to elements of the Rational Unified Process and the developers were using the Unified Modeling Language to describe system components. The refined rich picture as of September 2003 is shown in figure 5.3

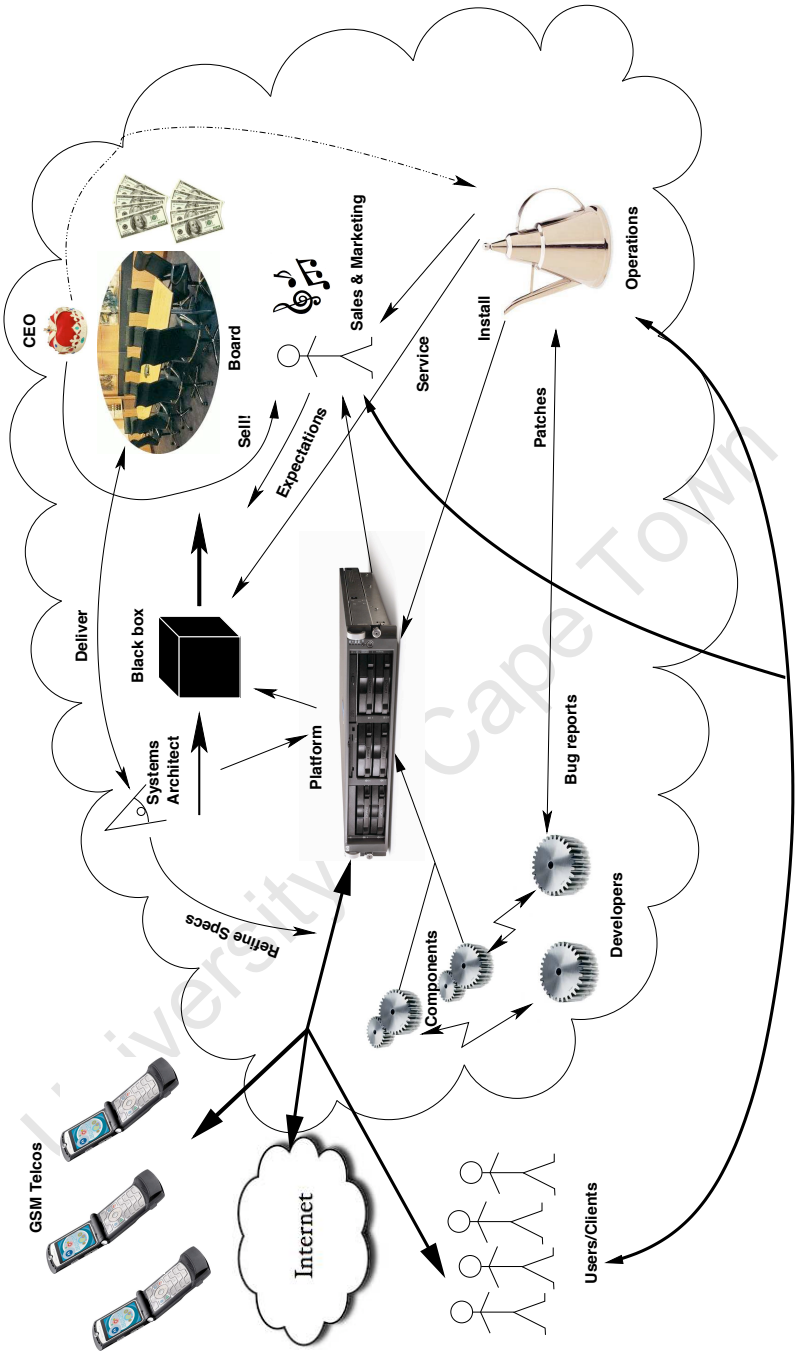


Figure 5.3: Rich Picture (as of September 2003)

The elements in this picture are the same as those illustrated in figure 5.1 and described in Table 5.1 but the rich picture is now a snap shot in time approximately

a year later. The development stage of the project was over and now the company was focusing on developing and running new services on top of the platform and fixing bugs as they were discovered. Certain elements were expanded to deal with the requisite variety found within the systems. The GSM telecommunication providers system for instance had been split into multiple providers. The reason for this was that not all the providers provided the same services nor did they all adhere to the necessary specifications, resulting in modifications that were needed in the platform to communicate with these providers.

Clients interacted with the company in two ways. Firstly they interacted with the sales and marketing departments, where their requirements were captured. These requirements were then fed to the operations department, whose task it was to manage the installation of the platform that met the clients requirement. Going forward the client interacted with operations in conjunction with sales who were responsible to service their additional requests, feedback to the developers any bugs discovered and apply patches to rectify these errors as well.

The new rich picture resulted in the root definitions being changed. The customer was no longer internal to the system, but external. The relevant CATWOE is outlined below:

C	The clients of the company purchasing the platform.
A	The Operations team.
T	The management of the process whereby the platform components were transformed into a operational communication platform by the operations team.
W	The *weltanschauung* was that of the high technology telecommunication information technology space.
O	The owner was the client purchasing the platform.
E	The emerging mobile telecommunication industry sector.

Not only had the environment, but also the client had changed to the emerging mobile telecommunication sector, but the transformation was chiefly around the management of the system construction. The configuration of the platform issues were dealt with by increasing the requisite variety of the system so that rather than needing to develop new code for each environment, the system was able to be

configured to meet the requirements of each site at which it was installed. From the new RD a new model was developed. This model has been illustrated according to figure 4.1, in figure 5.4.

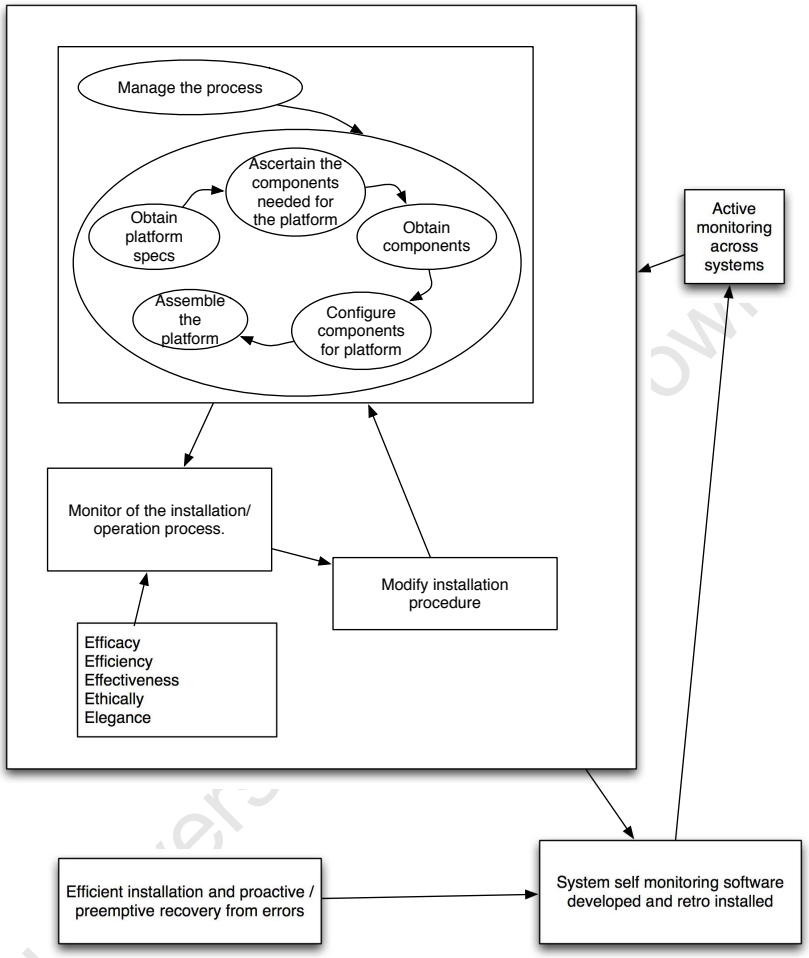


Figure 5.4: SSM Model - iteration number two

The main difference between this model showing in figure 5.4 and the previous model, shown in figure 5.2 was that the model had its effectiveness evaluated by the supra-system, normally controlled by the owner in the CATWOE. Effectiveness was defined as “Is this the right thing to be doing?” (Checkland, 2003). By taking into consideration the larger system the platforms boundaries could be managed more effectively.

A similar process to that which was carried out in the first iteration of the model was

undertaken. The model was evaluated and compared with reality. Similarities and differences were identified and changes that needed to be acted on were identified.

The new rich picture and the model were consistent with the platform and components, what was required and how to manage the configure and assemble them in such a way as to meet the clients requirements. The model now had its effectiveness evaluated not internally but rather by the owner of the system which was now external to both the system and the company.

The problems that occurred in this iteration were not directly related to the platforms configurations and code itself, but rather to external factors. These fell into two classes, firstly the platform needed to connect to telecommunication companies. Although the platform adhered to the relevant specifications for SMS communication protocols - SMPP, the telecommunications companies had taken some short cuts, because they assumed that no one would require these specific functions. In relying on this functionality, the platform functionality was broken. Secondly, and more importantly the marketing and sales teams were selling the platform from product sheets, which in some cases had not been updated to be consistent with what the platform was actually delivering. It then fell to the operations teams to mould the platform to do things it was not necessarily designed to do to meet the clients needs.

The main area missing from this was the impact of third party vendors and the lack of coherency between the product being marketed and that which was being installed.

Efficacy The model was internally consistent and worked.

Efficiency The method employed to construct the platform was efficient for this stage in production.

Effectiveness Done correctly the owner would have a functioning system and the system would effectively meet the long term goals of the client.

Ethically There were no ethical issues involved in this model.

Elegance If this model worked, it would streamline the process of the installation of the platform, and would be an elegant solution.

In order to resolve the identified issues two major steps where undertaken. Firstly on the technical side mobile phones from each of the clients countries mobile net-

work operators were obtained for testing and integration purposes. The software engineering methodology now employed to manage this project was heavily reliant on components of the RUP. Practically the idea was that these mobile phones would be used to do testing so that the idiosyncrasies of each of the telecommunication companies could be minimised and the platforms configuration could be adjusted accordingly. Secondly, it was decided to better manage the relationship between marketing and sales. This was done by highlighting the various products that were available on the platform and demonstrating their functionality to them in such a way that they could take the information provided and provide their clients with better information as to the capabilities of each product.

These action steps where undertaken and were evaluated in the following cycle.

5.3 The Third Iteration, in the UK in November 2003.

Despite changes propagated in iteration two of the SSM further issues were encountered, once again integration issues in the field created a problem. Back in the UK in 2003 the platforms failed to behave as they had done in the test environment in the companies offices in South Africa.

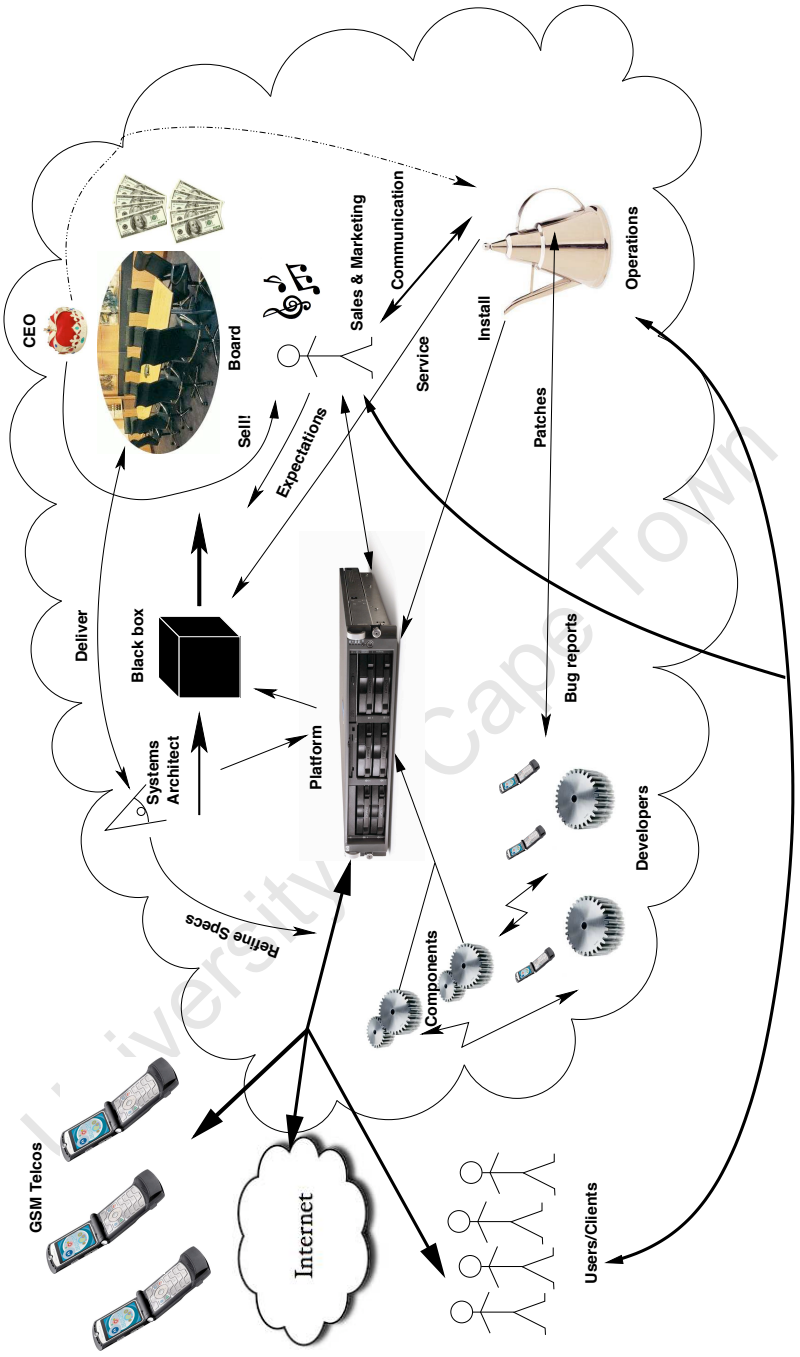


Figure 5.5: Rich Picture (November 2003)

In this iteration the importance of the boundaries of the system, that was modeled, were better understood. It was recognized that the platform was more than just the

sum of the components, but needed to be considered as part of the wider telecommunication environment and as such the external links need to be considered as part of the system as a whole.

The rich picture showed the mobile phones from the remote telecommunications providers where used in the unit testing phases of the assemble process. To ensure minimum time on-site, a custom lab that mimicked the clients environment was setup and using mobile phones from the clients home networks test and configurations where performed. Secondly, the interaction between operations and marketing and sales was strengthened by holding biweekly product focuses in which the marketing and sales department where exposed to the latest iterations of the various products on offer.

The CATWOE of this new rich picture remained unchanged from the last iteration:

C	The clients of the company purchasing the platform.
A	The Operations team.
T	The management of the process whereby the platform components were transformed into a operational communication platform *within a telecommunications environment* by the operations team.
W	The *weltanschauung* was that of the high technology telecommunication information technology space.
O	The owner was the client purchasing the platform.
E	The emerging mobile telecommunication industry sector.

It was assumed that the model that defined the transformation process, would be consistent with the real world. On examination it was found the model closely mapped to the real world and the changes proposed before should have been sufficient. What was not captured by the model were the details that although mobile phones from the clients mobile service providers were used, they were actually roaming on their South African partners networks. Tests undertaken here, showed only that the current system was working. The new aspects added to the Transformation, again modeled on figure 4.1, are shown in figure 5.6.

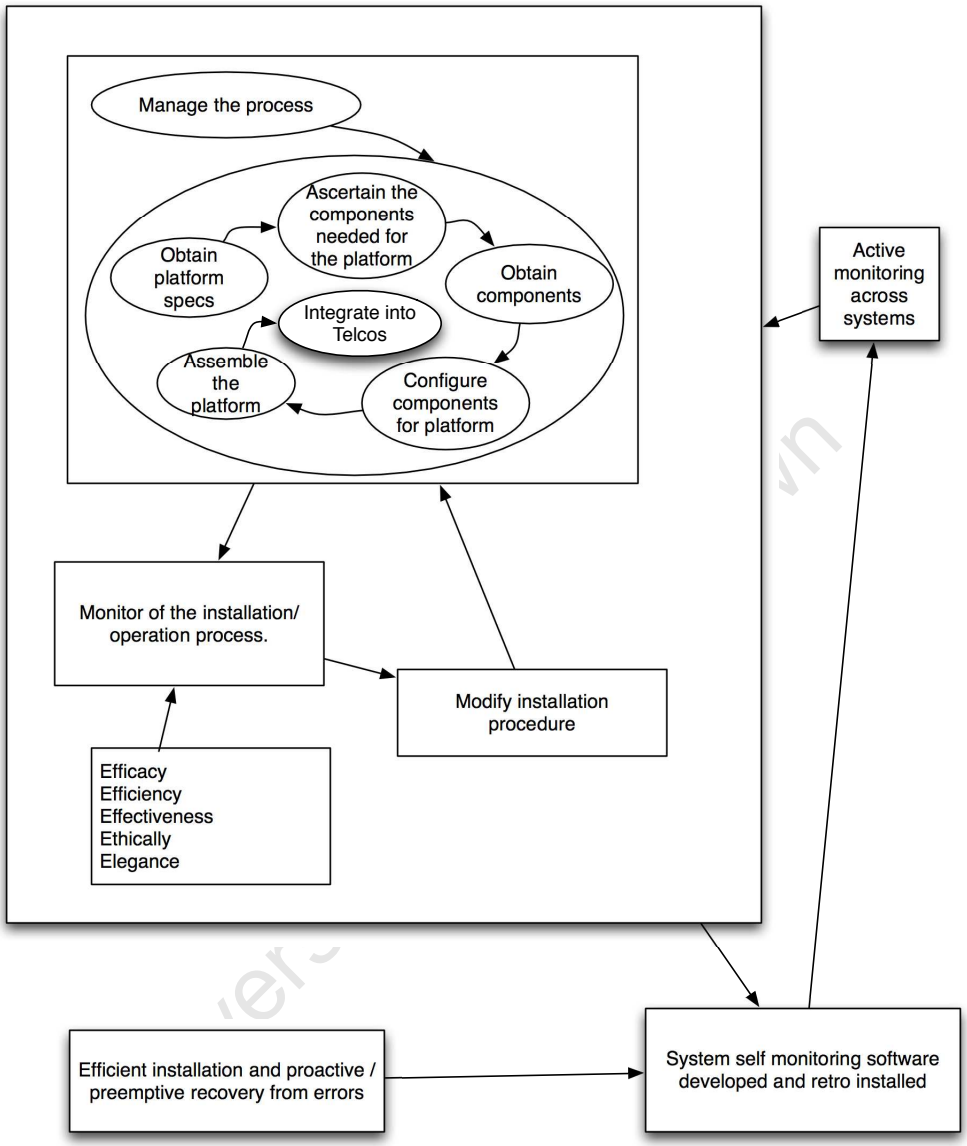


Figure 5.6: Third iteration conceptual model

The action that was undertaken was to comprehend the limits of the system that could effectively be managed. It was acknowledged that the boundary of the system were broader than in cycle one, but even though the boundary of the system in this cycle was included in the intervention stage, it was largely beyond the control of the system. In order to gain some control over this system a self-monitoring software solution was installed alongside the platform. This software had the properties that it was able to monitor the platform and the environment that it existed in and

alert the operators should it detect any errors. The combination of being aware of one's limits in the larger environment as well as having some automated method of determining if a variable was going beyond acceptable limits helped to effectively manage the costs of installing and operating the platform.

This chapter has demonstrated the action research process by applying the theoretical knowledge provided in previous chapters to a real problem, that of deploying a mobile communication platform in a number of different work environments. As each iteration was completed so the management methods were improved upon, thus modeling the process of action research. The conclusions to this experiment, as well as appropriate recommendations, are discussed in the following chapter.

Chapter 6

Evaluation and Conclusion

In this study an action research project with a systemic view of a mobile communications platform was undertaken. The concern of this study was to obtain a better understanding of which factors led to increased cost in software deployment. In order to answer this question it was shown that a suitable methodology to address this type of problem situation was to use a methodology developed by Checkland (1981, 1999) known as Soft Systems Methodology.

This methodology is one of the methodologies of action research. By examining and justifying a constructivist view of knowledge this study was able to show that in order to understand the complexity of the real world, the perception of the world as a construct needs to be acknowledged. The problem formulation strategy used showed that there are a class of real world problems that are not able to be neatly classified. Known as messy problems these problems are best addressed by using what Ackoff (1981) calls the design approach to "dissolve" them and so manage the problem.

The situation in which this concern arose was in an area of information technology that is governed by software engineering. The different systems engineering methodologies were examined from both a historical perspective: looking at the waterfall method and a modern perspective: examining and looking at two of the current methodologies, Agile development and the Rational Unified Process. The methodology that was chosen as the most suitable to manage this project emerged as the Rational Unified Process. The methodology had the best fit to the problem raised by the research question. Hull, Taylor, Hanna, and Millar (2002) acknowledged that software development and maintenance have become a major challenge.

They examined the best practices that have emerged in dealing with these problems over the last decade. From a management perspective they showed the Ration Unified Process model to be the best fit to managing this types of issues.

Using Action Research in the application involved planning an action, then deciding what the intention was. As part of the system implementing that intention involved doing the action and observing critically what effects they have on the system in focus. The next step was to critically evaluate the results of the action to determine what had happened. This examination was done critically. That means looking both for what one expects to happen because of predictions and also what actually happened, that was not expected. The latter was actually more helpful as it showed the gaps in the existing knowledge. The process was cyclical so that the newly gained knowledge could be refined and the practitioner was to undertake more relevant action.

6.1 Overview of action taken

In the study a number of cycles where undertaken and these corresponded roughly to the number of off-site installations. Changes were made incrementally so as to reduce the risk to the company should they fail. The purpose of each action was to better improve the management of the deployment and maintenance of the platform. These cycles where coupled with SSM in order to make sense of the real world situation and add rigour to the investigation process.

After the first on-site installation a post-installation meeting was held in order to reflect on what had been learnt by the company. The consensus at this stage was that as this was the first international installation it had gone well, as ultimately the product worked to the clients specification. This despite the fact that it had overrun time significantly and had taken close to two weeks to get installed and configured in the clients hosted environment. Technical areas concerning the lack of documentation and configuration parameters were addressed. It was hoped that these would be rectified in the coming months.

Almost a year passed in which development of the platform flourished before returning once again to Tanzania to complete a second major installation. Although this was a return to the same environment in which the company had been managing the existing platform the addition of new services proved more difficult than

had been expected. The original software was well contained and was a rather simple application. These new products were of a higher complexity and required a lot more configuration and interoperability. It was soon discovered that not only where sensible defaults not used, but that the developers has left many sections of the configuration parameters undocumented and not exposed to configuration. On returning home the management of the platforms configuration, deployment and testing phases where examined. In order for the developer to produce quality code that was able to perform beyond its development environment, a clean room firewalled network was set up. This network environment could mimic the clients environment and the full configuration of the platform was testing within this environment prior to departure. Change control was also introduced and it was required of the developers to perform full integration testing and unit testing before committing new code to the platform

The following installation was in the United Kingdom into a different hosted environment. Whereas the previous installations had been directly in a mobile telecommunications company and connecting to only one SMSC, these next installations were external to the telecommunications company and required the use of either third party client (WASP) to connect to the SMSC or leased lines over the net connecting to the SMSC. It was soon discovered that although these WASP claimed to support the SMPP specification, they in fact only supported a commonly used subset of the specification. Full compliance was required as the platform did not have the necessary requisite variety to cope with non-standing interfaces. This resulted in total failure of some of the products, which resulted in a major rewrite of the system to accommodate these idiosyncrasies. These changes were not fully enabled and the team returned home with a new set of management requirements for the platform and testing.

The requirements for the management of the deployment of the platform and its configuration had changed. Not only was it required that the platform accommodate the new non-standard methods of connection, but some clients required external monitoring software to ensure the platform was working effectively. The bulk of the work around the management of the platform was carried out at that stage. The following changes were made to the management methods employed: The system was deployed and tested in the clean-room firewalled environment with mobiles phones that had connections to each of the clients service providers. The purpose of this was to ensure that the code worked on the client home mobile net-

work and not necessarily just the test ones. Soon it became evident that although the mobile phones of the remote network were being used, because these where geographically located in South Africa, they were just being tested on the already proven local mobile communication companies network with which the remote mobile providers had roaming agreements. The net result was that the team resorted to using mobiles phones in the UK to test the platform hosted in South Africa via the Internet. The second major change in management was that an automated customizing processes was developed that would take the necessary variables and install them into all the required component configuration files as well as install the require components and its dependencies. Lastly a monitoring system was developed that gave operators a look into the 'health' of the system. It monitored the underlying hardware, third party software and even reported errors in the platform pro-actively to the operator via sms and email so that they where fully informed as to the state of the platform.

These changes were implemented over the next few installations of the platform, which are beyond the time frame of this research but were also retroactively fitted to the existing platforms. In comparison the first installation for one platform and a backup in 2001 which took close to two weeks. It involved three engineers on-site and a team back home to get fully functioning. The last installation in February 2005 of three platforms and two backup platforms took approximately three hours each to install and required two engineers on-site for a day and a half to get them integrated, tested and fully functional. The majority of that time was spent in the data centers connecting the systems to the existing network and configuring the necessary off-site connections required for these platforms to be operational. These platforms were fully self monitoring and alerted the operators should any faults be detected both within the platforms themselves and also if they detected any loss of connectivity to the hosted environment for example. Should the network fail or connection be lost to the various SMSCs then an engineer received notification. Although the platform did not have control over the system in which it was hosted, it was able to manage the situation so that one could be aware of its operations status at any time.

6.2 Reflections and suggestions

The management of these complex platforms, which consisted of over two million lines of Java code, came a long way and the understanding of the factors contributing to the initial concern were improved. Rose (2002) acknowledged the fitness of SSM to information system development problems. He examined the software engineering model used by Checkland and concluded that the traditional waterfall methodology he employed did not have a good fit with the systemic approach that the rest of SSM brings to the problem space. He proposed a new model of software development, which was more iterative and showed that the model views system development as a process that is more managerial and social than technical. Hull et al. (2002) also showed that the development process was a managerial component and this research chose to use a combination of SSM and RUP to address the concern of how to best manage the software engineering process so as to maximize quality and customer satisfaction and at the same time reduce installation and operational costs.

When examined the action research paradigm literature tells us that 'this method is widely cited as a exemplar of post-positivist social scientific research method, ideally suited to the study of technology in human context" (Baskerville and Wood-Harper, 1996, pp 235) and in particular "Checkland's soft systems methodology has influenced IS research by linking action research and systems development" (Baskerville and Wood-Harper, 1996, pp 235). They summarize "action research is a method that could be described as a paragon of the post-positivist research methods. It is empirical, yet interpretive. It is experimental, yet multivariate. It is observational, yet interventionist" Baskerville and Wood-Harper (1996, pp 236).

Acording to Baskerville and Wood-Harper (1996) action research methods have three distinctive characteristics, these are:

1. "The researcher is actively involved, with expected benefit for both researcher and organization,
2. the knowledge obtained can be immediately applied, there is not the sense of the detached observer, but that of an active participant wishing to utilize any new knowledge based on an explicit, clear conceptual framework,
3. the research is a typically cyclical process linking theory and practice" (Baskerville and Wood-Harper, 1996, pp 239)

This study had all these elements present. The researcher was employed in this organization and the work undertaken benefited both the company as well as the researchers learning. The knowledge gained in this study was immediately ploughed straight back into the companies projects and the learning's were used to feed into the next cycle of development within the company. On reflection one of the difficulties experienced in this process was getting the entire management team to embrace the method of action research. As the company had a large number or directors for its size, each was building their own area of expertise. Within the Operations and Development section, there was a more casual community of people who met more formally weekly to discuss the on going development and issues that arose, as well as a number of more informal meetings between the heads of the various teams to tackle more directly the issues as they arose. These informal meeting formed a large component of the data that was used within this study, but by its very nature was not minuted in a formal way. This data was captured on whiteboards and flip charts, and distilled into notes that formed that data of this project. Two issues arise from this type of research , firstly that the process lacks rigour and secondly that an "action research project, by nature of its intervention into a unique organizational setting, can never be repeated" (Baskerville and Wood-Harper, 1996, pp 243) and hence its outcome cannot be generalized.

Concerning rigour, "the lack of impartiality of the researcher has led to rejection of the action research method by a number of researchers. However this is not necessarily a problem singular to the action research method. It is rooted in the philosophical supremacy of the researchers. Philosophical supremacy refers to the refusal of scientists to accept any knowledge founded in any alternative philosophy of science other than their own" (Baskerville and Wood-Harper, 1996, pp 240).

Baskerville and Wood-Harper (1996, pp 241) quoting Straub 1991 indicated that "rigour relates to fitting the research methods to the problem in order to produce valid scientific explanations, and the use of multiple methods to produce valid research constructs." In this study, rigour in the research has been achieved in a number of ways. Firstly, by using a well known and rigorous methodology, SSM. Secondly, using many action research cycles of intend, act and review. Baskerville and Wood-Harper (1996, pp 241) state "rigorous action research clings tenaciously to its disciplined constructs of cyclical theoretical infrastructure, data collection and evaluation: there is a clear cycle of activity; there is a premise; a pronounced theory (under test); there is empirical data collection (for example, diaries)." When

we consider the issue of rigour Baskerville and Wood-Harper (1996, pp 241) conclude that "these problems are actually general problems of social science research. In reality action research shares these problems with the other methods. Perhaps the distinguishing difficulties with action research are those of degree rather than taxonomy".

A second issue that arises is how can this research be generalized. "Action research, being naturally idiographic, presents researchers with a serious conflict regarding any generalization from the project findings. As mentioned earlier, only the most tentative causal links can be claimed owing to the multivariate nature of the study. Yet for a vocationally-oriented field like IS, it is the promise of generalizability that interests the majority of scientists" (Baskerville and Wood-Harper, 1996, pp 243). How do we reconcile the idiographic nature of action research and the promise of generalizability? The issue of the management of the operational challenges with respect to a mobile communications platform was examined. This occurred within a specific organization, at a specific, non repeatable time. Can one generalise the findings of this study to the class of issues that arise from these types of situations? Baskerville and Wood-Harper (1996, pp 243) state that "action researchers can legitimately generalize their findings on the basis of the validity of their research. In addition, action researchers can design synchronic reliability into the structure of their research project. However they must exercise restraint in their conclusions since these must be reported from a limited number of observations. This, of course, implies another characteristic of rigorous action research: circulation of the results to the scientific community"

In summary Baskerville and Wood-Harper (1996, pp 244) concluded that "action researchers can achieve scientific rigour through a number of characteristic strategies. First they must establish an ethical client-system infrastructure and research environment. They must plan their data collection carefully. They must observe iterative phases that formulate theory, plan action, take action, and evaluate the action. Through this process they must promote collaboration by the subjects and support their subjects' learning cycles. Despite the idiographic nature of the study, the researcher may imply certain generalizations based on the theory and learning."

The question raised in this study was which factors led to increased costs in operational and software deployment of the mobile communication platform? From the software development perspective what emerged was that the software development methodology has dramatic effects on the success of the outcome of the

project. Failing to consider the broader deployment environment and not having a methodology that can adapt quickly to the changing environment can hamper the successful deployment of the platform. From an operational perspective, the deployment of the platform needs to take into account the system in which it is deployed, and the management between the surrounding systems needs careful planning. Firstly, it needs to be recognised that when deploying such a platform that a large majority of the systems to which the platform connects to are vital to functioning of the platform and are beyond operational control. These interfaces between systems need careful management. The second area is that of change control. Developers changes need to be interrogated and tested in situ so as to confirm that they take cognisance of the environment in which they are running. By defining and managing boundaries both within the platform and the environment in which it was deployed, we can reduce the cost of both development and deployment.

Finally the research question for this thesis was "How does the lack of knowledge about the management of the software engineering methodology employed by management team and deployed by the developers affect the operational deployment costs of the platform?" In the context of operational management this was defined as a messy problem. Within the action research paradigm, Soft Systems Methodology was determined the most suitable methodology to address the research question. The result of the study determined that by controlling two factors, firstly the elements within the platform and secondly the environment without the platform would have significant consequences in reducing the costs of the deployment and management of the platform. This being said, in general the knowledge gained in resolving the Research Question for this study can be applied to this class of Information Technology problems.

Bibliography

A. Abran, J. W. Moore, P. Bourque, R. Dupuis, and L. L. Tripp, editors. *Guide to the Software Engineering Body of Knowledge - 2004 Version.* IEEE, 2004.

R. L. Ackoff. The Art and Science of Mess Management. *Interfaces*, 11(1):20–26, February 1981.

M. Akmanligila and P. C. Palviab. Strategies for Global Information Systems Development Strategies for Global Information Systems Development Strategies for Global Information Systems Development Strategies for Global Information Systems Development. *Information & Management*, (42):45–59, 2004.

H. Altricter, S. Kemmis, R. McTaggart, and O. Zuber-Skerritt. The Concept of Action Research. *The Learning Organization*, 9(3):125–131, 2002.

R. Baskerville. Investigating Information Systems with Action Research. *Communications of AIS*, 2, October 1999.

R. L. Baskerville and A. T. Wood-Harper. A critical perspective of action reasearch as a method for information systems research. *Journal of Information Technology*, 11:235–246, 1996.

K. Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.

K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, 2001 `http://agilemanifesto.org/`.

P. D. C. Bennetts, A. T. Wood-Harper, and S. Mills. An Holistic Approach to the Management of Information Systems Development - A View Using a Soft

Systems Approach and Multiple Viewpoints. *Systemic Practice and Action Research*, 13(2):189–205, 2000.

B. Bergvall-Kareborn, A. Mirijamdotter, and A. Basden. Basic Principles of SSM Modeling: An Examination of CATWOE From a Soft Perspective. *Systemic Practice and Action Research*, 17(2):55–73, 2004.

W. C. Booth, G. G. Colomb, and J. M. Williams. *The Craft of Research*. The University of Chicago Press, first edition, 1995.

F. Brooks. *The Mythical Man-Month: Essays on Software Engineering, 20th Anniversary Edition*. Addison-Wesley, 1995.

F. P. Brooks. No Silver Bullet: Essence and Accidents of Software Engineering. *Computer*, 20(4):10–19, April 1987.

P. Checkland. Soft Systems Methodolgy, May 2003 `http://www.onesixsigma.com/_lit/white_paper/peterchecklandssm.pdf` Last visited: 30 December 2005.

P. B. Checkland. *Systems Thinking, Systems Practice.* John Wiley and Sons Ltd, 1981.

P. B. Checkland. *Systems Thinking, Systems Practice.* John Wiley and Sons Ltd, 1999.

P. B. Checkland. OR and Systems Movement: Mappings and Conflicts. *The Journal of the Operational Research Society*, 34(8):661–672, 1983.

P. B. Checkland and S. Holwell. *Information, Systems and Information Systems.* John Wiley and Sons Ltd, 1998.

P. B. Checkland and J. Scholes. *Soft Systems Methodology in Action.* John Wiley and Sons Ltd, 1999.

L. Cohen, L. Manion, and K. Morrison. *Research Methods in Education*. RoutledgeFlamer, 2000.

M. Crotty. *The Foundations of Social Research*. Sage Publications Inc, 1998.

B. Dick. Action Learning and Action Research, 1997 `http://www.scu.edu.au/schools/gcm/ar/arp/actlearn.html` Last visited: November 2007.

B. Dick. You Want to Do an Action Research Thesis?, 1993 http://www.scu.edu.au/schools/gcm/ar/art/arthesis.html Last visited: November 2007.

B. Dick. What is Action Research?, 1999 http://www.scu.edu.au/schools/gcm/ar/whatisar.html Last visited: November 2006.

A. Drobik. Dot-Coms: Is the Phoenix Rising? Research Note: M-14-2782, October 2001.

K. Ewusi-Mensah. Critical Issues in Abandoned Information Systems Development Projects. *Communications of the ACM*, 40(9):74–80, 1997.

R. L. Flood. A Brief Review of Peter B. Checkland's Contribution to Systemic Thinking. *Systemic Practice and Action Research*, 13(6):723–731, 2000.

R. L. Flood and M. C. Jackson. *Creative Problem Solving: Total System Intervention*. John Wiley and Sons Ltd, 1991.

M. Fowler and J. Highsmith. The Agile Manifesto. *Software Development*, August 2001.

S. Holwell. Soft Systems Methodology: Other Voices. *Systemic Practice and Action Research*, 6(13):773–797, 2000.

M. Hull and S. A. Lennung. Towards a definition of action research: a note and bibliography. *Journal of Managment Studies*, 17:241–250, 1980.

M. Hull, P. Taylor, J. Hanna, and R. J. Millar. Software Developement Process - an Assessment, 2002.

D. Jackson. Dependable Software by Design., 2006 http://www.sciam.com/article.cfm?articleID=00020D04-CFD8-146C-8D8D83414B7F0000&sc=I100322 Last visited: December 2007.

M. C. Jackson. *Sytem Approaches to Managerment*. Kluwer Academics/Plenum Publishers, 2000.

K. Johnson. Denver Airport Saw the Future. It Didn't Work. *New York Times*, 27 August 2005.

J. F. Kennedy, 1962 http://www.historyplace.com/speeches/jfk-space.htm Last visited: August 2006.

P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley Professional, 3rd edition, 2003.

P. Kruchten. What is the Rational Unified Process?, 2001 http://www.therationaledge.com/content/jan_01/f_rup_pk.html Last visited: November 2006.

V. S. Lai. Issues of International Information Systems Management: a Perspective of Affiliates. *Information & Management*, (38):253–264, 2001.

P. A. N. Lars Mathiassen, Andreas Munk-Madsen and J. Stage. Soft Systems in Software Design. *Systems Thinking in Europe*, 1991.

A. Ljungqvist and W. J. Wilhelm Jr. IPO Pricing in the Dot-com Bubble. *The Journal of Finance*, LVIII(2), 2003.

M. Marchesi. The New XP, 2005 http://www.agilexp.org/downloads/TheNewXP.pdf Last visited: November 2006.

E. Maxson. Soft Systems Methodology CATWOE Criteria, Fall 2005 http://cispom.boisestate.edu/cis310emaxson/catwoe.htm Last visited: June 2006.

G. Midgley. *Systemic Intervention*. Springer, 2001.

G. Miller. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *Psychological Review*, 63:81–97, 1956 http://psychclassics.yorku.ca/Miller/ Last visited: December 2007.

G. E. Moore. Cramming more components onto integrated circuits, 1965 http://download.intel.com/museum/Moores_Law/Articles-Press_Releases/Gordon_Moore_1965_Article.pdf Last visited: December 2007.

C. Mouly. *Educational Research : the Art and Science of Investigation*. Boston: Alleyn and Bacon, 1978.

C. S. Peirce. How to Make Our Ideas Clear, 1878 http://www.peirce.org/writings/p119.html Last visited: March 2007.

K. Popper. *Conjectures and Refutations: The Growth of Scientific Knowledge*. Routledge and K. Paul, 1963.

J. Rose. Interaction, Transformation and Information Systems Developement - an Extended Application of Soft Systems Methodology. *Information, Technology & People*, 15(3):242–268, 2002.

T. Ryan. A Framework for an Action Learning and Research Project. Executive MBA, Graduate School of Business, University of Cape Town, 2005.

I. Sommerville. *Software Engineering, 6th Edition*. Pearson Education (Addison Wesley), 2001.

The Rational Edge. The Ten Essentials of RUP `http://www.devarticles.com/c/a/Development-Cycles/The-Ten-Essentials-of-RUP/`.

G. Walsham. The Emergence of Interpretivism in IS Research. *Information Systems Research*, 6(4):376–394, 12 1995.

A. Waring. *Practical System Thinking*. Thomson Learning, 1996.

C. H. Yu. Abduction? Deduction? Induction? Is there a Logic of Exploratory Data Analysis? In *Annual Meeting of American Educational Research Association*. Annual Meeting of American Educational Research Association, 1994.

O. Zuber-Skerritt. *Professional Development in Higher Education. A Theoretical Framework for Action Research*. Routledge, 1995.

Appendix A

Documentation Relevant to the Practical Problem

A.1 Document One

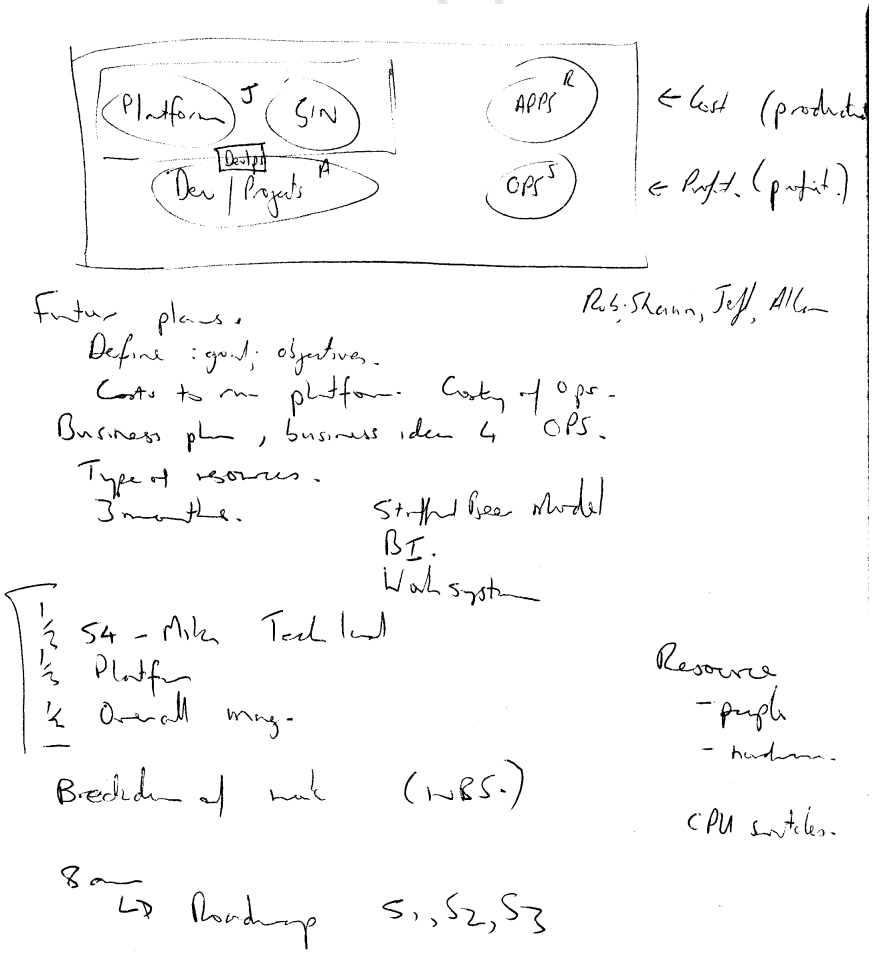


Figure A.1: Early planning meeting, November 2001

A.2 Document Two

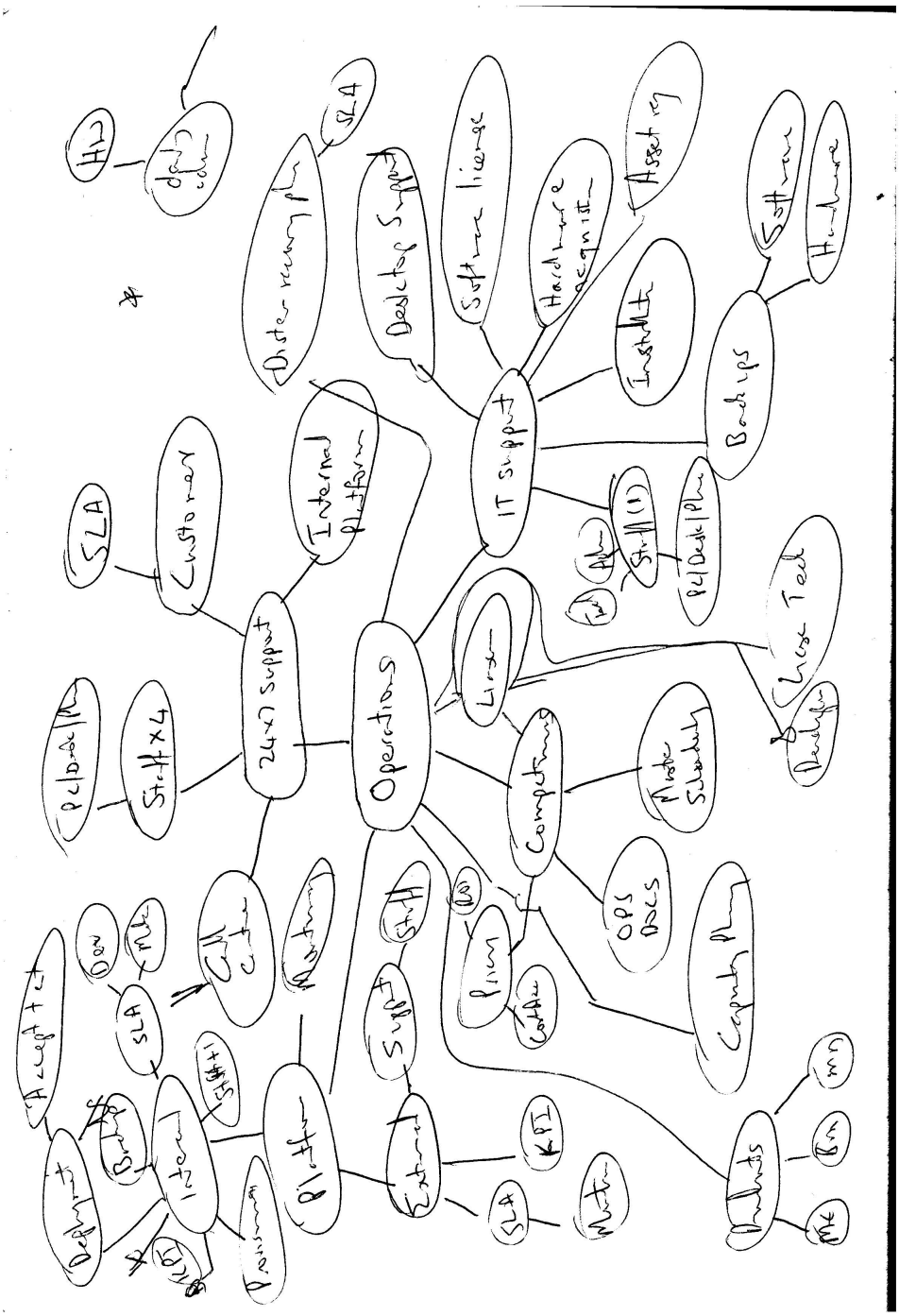


Figure A.2: Mind Map of Operation System, Early 2002

A.3 Document Three

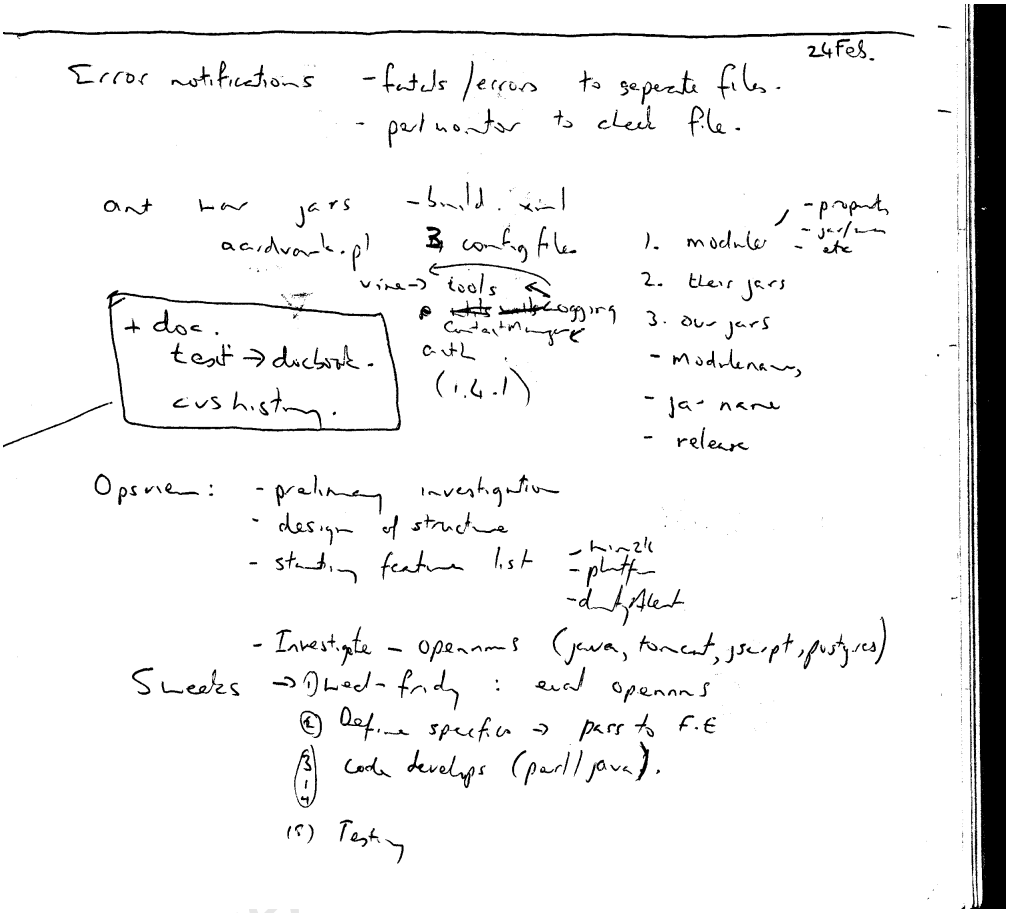


Figure A.3: Sample of Meeting Notes 24 February 2003

A.4 Document Four

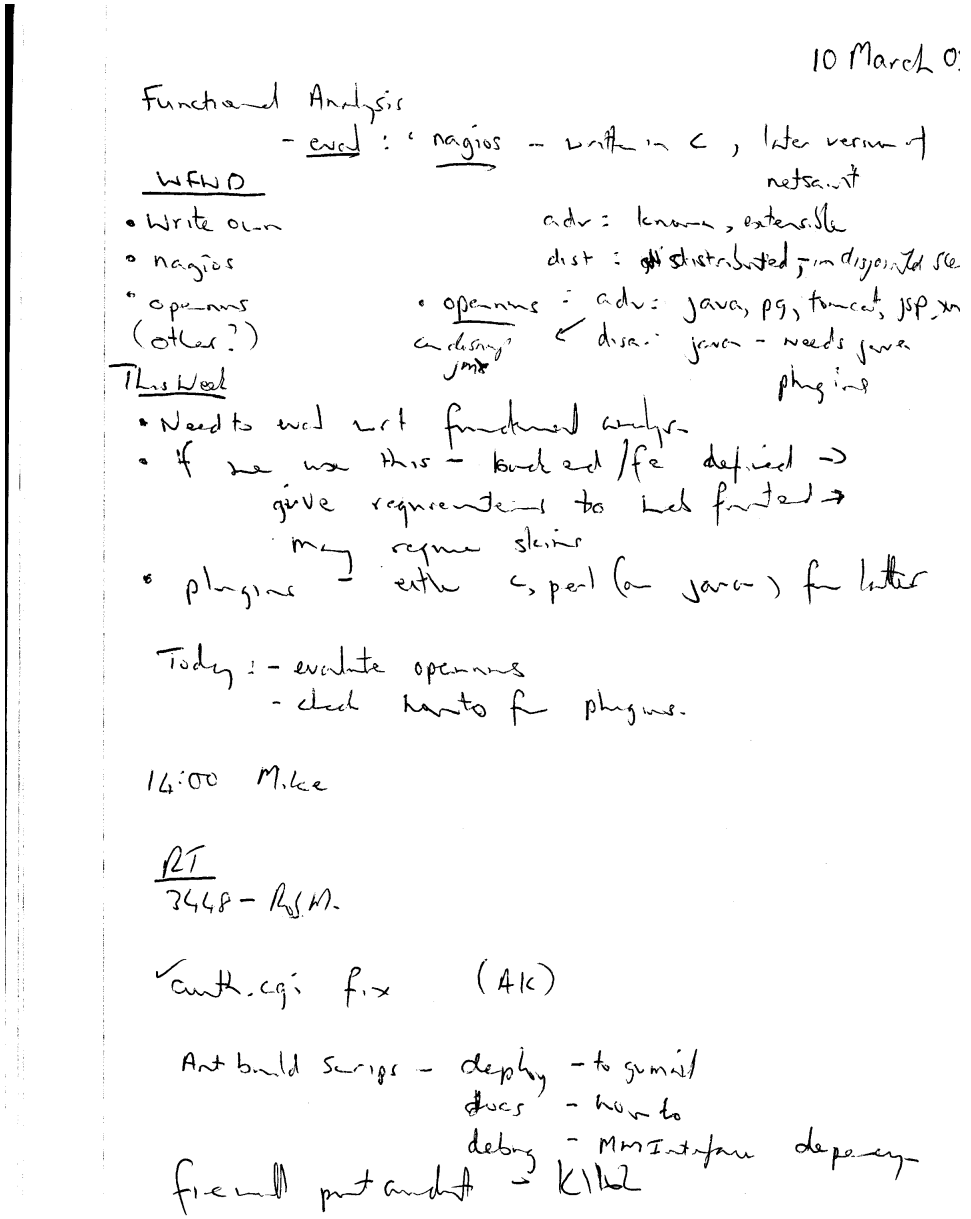


Figure A.4: Functional Decomposition notes 10 Mach 2003

A.5 Document Five

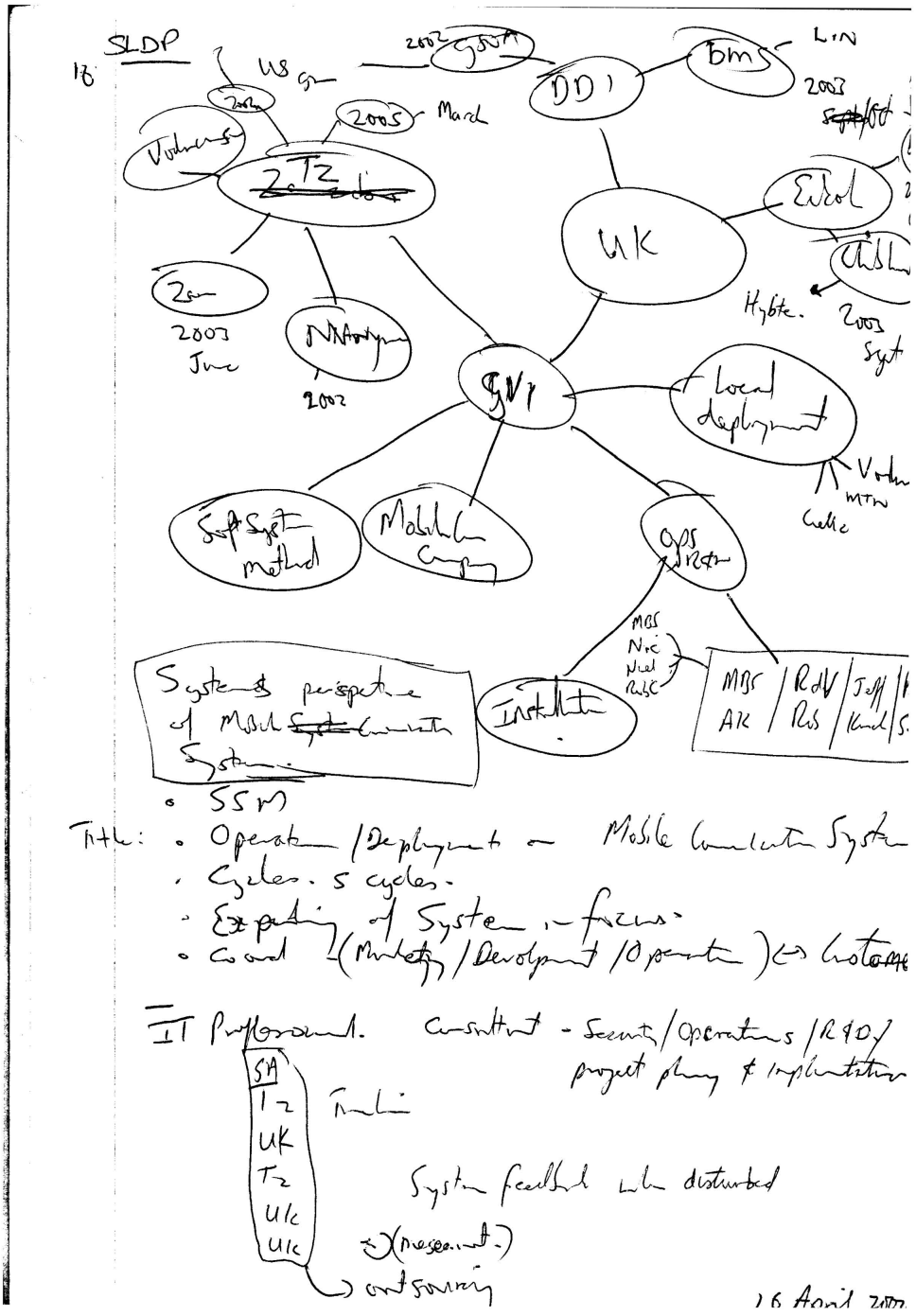


Figure A.5: Planning notes 16 April 2004

A.6 Document Six

This is a copy of the document explaining the way forward for the Operations team.

Operational Challenges 2003

Shaun Courtney

Here are some of the challenges facing Operations over the next few months:

Some background.

The Company has chiefly been a development house and therefore platforms have been run up until August 2002 exclusively by the developers. This has both advantages and disadvantages. I will outline some of these as well as some suggestions for working around these problems.

Firstly the advantages: Numbering around 12 - 14 people make the load of running the production servers was minimal. Offering 24/7 support was possible is it meant someone would be on duty (18:00 to 8:00) once every two weeks. Another advantage was those working on they system were intimately familiar with them (as they developed them) and therefore required little or no documentation on both how the system as a whole was integrated and worked as well as how to configure it. Problems were resolved relatively quickly as there was a large pool or resources to call on the fix and serious problems.

Secondly the disadvantages: About six months ago we started moving away from the model where the developer maintained the production platforms and the job was handed over to Aubrey (Ops at that stage). Over time the knowledge of the systems have moved from the developers to operations. This has brought some unique problems. One is that the Ops team were not involved in development of the platforms nor their configuration. As little documentation on the overall platform or configuration of the specific components exist the Ops team relies on the developers for help on how to install and run various aspects of the platform. Lack of front end components required from the Ops perspective, means adding new clients requires access both to the front and back-end. (Acceptable for the Company staff, but definitely a problem for 3rd parties who purchase the platforms.) Another disadvantage of Ops running the platforms is the number of staff has been

effectively reduced from 14 or so to 2. This means that Ops staff are working overtime (18:00 - 08:00) every second day. Effectively working the nominal 8 hours a day and the 16 hours on standby. This means Ops staff work at least 40 hours a week and then 54 hours overtime a week on average.

Going forward.

Given this background Operations is facing a new round of challenges. With the segregation of the departments within the Company in R&D, Projects and Operations, we are suffering from a severe shortage of human resources. With all production systems moving from the developers to Operations there are a number of key areas that need to be addressed.

Currently Ops has two staff members. Besides working and being on call 104 hours a week we are now faced with the challenges of running the junior system, the database servers, production servers, finance queries, IT support for the Company staff, maintenance and upgrades of the systems, disaster recovery as well as 24/7 customer support, competitions and running projects to patch, upgrade and maintain both the Company as well as growing number of clients: Tanzania, GSOA DiData, Guinness and ITouch. Currently there are about 15 production machines, and these need to be monitored manually currently. Over and above these tasks we are faced with an increasing number of SLA's that have been signed.

Current SLA's recommend that Severity 3 problems (the lowest) should be solved within 8 work hours or 12 after hour hours. Severity 1 errors to be fixed in 4 hours. Should this not be done within 100% of the time the Operations Manager is called so that the call can be escalated. A scenario of a problem occurring at 17:00 in London (19:00 SAST) - by 23:00 it should be fixed, by 03:00 in the morning the Ops Manager is called to escalate. The problem is there are no staff to escalate the problems to. Other items listing in the SLA is the speed of delivery. In order for us to have any idea how to manage this we need to first measure these indicators. (another job for Ops)

Another challenge for Operations is taking over the database maintenance and running of the junior system. None of the current Ops staff are DBAs and really are not qualified nor able from a time perspective to manage and administer the database machines. With the separation of developers from ops we have been ham-strung in

performing even the simplest queries. Should we take over the junior system, currently run and maintained by 2 people will further burden the Ops department. The nature of these systems required extremely slow access via PC Anywhere to manage and requires some knowledge of MS SQL, visual basic and the mail program to do traces of the queries received.

Suggestions.

If the Company wants to progress and move the running of is productions server by Operations and enforce a separation from the development team a few items need to be addressed. Personally I feel that this is the way that the Company needs to move, but it will need the support of the management team.

The Company's operations are going global (Oz, Israel, Tanzania, SA, Mozambique, UK) and as we deploy more platforms and run more services, Hunters, Vodacom soccer, Guinness, Ttouch, Investec we are going to be bound by our SLA.

Firstly if we are to offer 24/7 support we need to increase the resources. We need at least 4 people to work on shift in a call center to offer first line support as well as adding and removing clients and services. A week has 168 hours, with the work week of a person being about 40 hours, so 4 x 40 = 160. The missing 8 hours can be covered by existing staff.

Secondly we need a database administrator to maintain, backup, tune and run queries on the databases the we run on the various platforms. Currently we have 6 production databases that need to be maintained.

Thirdly we need developers on call 24/7. If the SLA calls for 4 hour at least and 12 hours as max, resolution on calls we need to have people who are familiar with all aspects of the system to provide support.

Fourthly we need technical, user and configuration documentation on each aspect of the platform. The technical documentation will be needed by the Ops staff that do installations and bugfixs to the platform. The configuration docs will be used to configure for various services that run and the user documentation will be required by the first level support team in supporting customers.

This should free the existing Operations staff to handle second level support, setup custom competitions, project manage deployment of new platforms and the instal-

lation of bugfixes. They would also handle new deployments and the maintenance of the existing platforms, IT support and overall management of Operations.