The deployment of systems development methodologies at project level in software houses in South Africa

B.D. Janse van Rensburg Hons. B.Com.

Dissertation submitted in partial fulfillment of the requirements for the degree *Master of Commerce* at the Potchefstroom Campus of the North-West University

Supervisor: Prof. H.M. Huisman

November 2006

Acknowledgements

First of all I would like to thank God for giving me opportunities to grow, and for surrounding me with people who carry me when I fall.

Dr. Huisman, my supervisor, motivator and friend, thank you very much for all your late nights and early mornings. I would also like to thank Rizelle and Estie for always being there for me, for motivating me and helping me; my family for all their prayers and support; and all of my friends for believing in me.

Abstract

This is a study of the deployment of systems development methodologies at project level in software houses in South Africa. This study extends previous research of Huisman and Iivari (2002a, 2002b) who studied the deployment of system development methodologies at organisational level and at individual level. More specifically, the author studied the context and the outsourcing environment in which system development takes place in software houses in South Africa. Furthermore, the use of systems development methodologies at project level in software houses in South Africa and the perceived support that the systems development methodologies provide were studied.

In this study the author used a qualitative research method. Multiple case studies were done and the data were collected using semi-structured interviews. These interviews were transcribed and a cross-case content analysis was done with the help of ATLAS/ti.

Software houses are increasingly important in the information system development field, especially when taking into account the movement to external development and outsourcing. The constantly changing user requirements as well as the changing environment cause many challenges for the projects in South African software houses. These challenges can be categorised into user related challenges and environment related challenges. The user related challenges include: changing user requirements, managing client expectations, availability of the users and resistance to change. The environment related challenges includes: the constantly changing environment, the difficulty of integrating with 3rd party systems, the neglecting of change management and the lack of readiness for change in the client organisations' culture.

It was found that all the projects used systems development methodologies. The projects that involved enterprise resource planning systems all followed the ASAP methodology. The other projects used in-house methodologies based on different parts of the object oriented as well as the extreme programming methodologies. These methodologies are not used strictly by the software houses; instead they are used in a flexible manner to handle the changes and challenges that may arise during a project.

The support provided by the use of systems development methodologies includes control support, cognitive- and cooperative support and production support.

Support as control technology reflects project management support, ways of tracking progress and keeping to a schedule and budget. It was found that a systems development methodology gives structure and guidance to the project and improves accountability. Cognitive and cooperative support illustrates the way the methodology guides the way team members work together, and the influence it has on communication and the exchange of information. In this study it was found that the use of systems development methodologies improves documentation and communication, this includes communication between team members as well as the communication between the software house and the client organisations. Support as production technology provides the project team with tools, techniques and methods to help them with the development. The general feeling that surfaced was that requirement analysis is a difficult but very important part of the system development process. It was found that many of the cases used methodology based tools to improve the requirement analysis process.

The perceptions of project team members on the use of system development methodologies showed that the use of systems development methodologies results in a unique improvement in the areas of communication, documentation and accountability. These are important advantages, especially for software houses, where client relationships and knowledge management are vital.

Opsomming

In hierdie studie word die gebruik van stelselsontwikkelingsmetodologieë op projekvlak in Suid-Afrikaanse sagtewarehuise ondersoek. Die studie brei uit op vorige navorsing deur Huisman en Iivari (2002a, 2002b) wat reeds die gebruik van stelselsontwikkelingsmetodologie op organisasie en individuele vlak bestudeer het. Meer spesifiek bestudeer hierdie studie die konteks en uitkontrakterings omgewing waarin die ontwikkeling van stelsels in die sagtewarehuise plaasvind. Verder word daar gekyk na die gebruik van die stelselsontwikkelingsmetodologie asook die ondersteuning wat deur die gebruik daarvan verleen word.

Tydens hierdie studie is gebruik gemaak van 'n kwalitatiewe navorsingsmetode. Meervoudige gevallestudies is uitgevoer en die data is verkry met behulp van semi-gestruktureerde onderhoude. Hierdie onderhoude is getranskribeer waarna die inhoud met behulp van 'n kruis-geval analise ondersoek is, Die groepering van soortgelyke inligting is met die hulp van ATLAS/ti. sagteware gedoen.

Sagtewarehuise raak al hoe belangriker in die stelselontwikkelings veld, veral as mens die beweging na eksterne ontwikkeling in ag neem. Die omgewing wat verander asook die gebruikers behoeftes wat gedurig verander veroorsaak verskeie uitdagings vir die sagtewarehuise in Suid-Afrika. Hierdie uitdagings kan verdeel word in gebruiker verwante uitdagings en omgewing verwante uitdagings. Die gebruiker verwante uitdagings sluit die volgende in: verandering in gebruikers behoeftes, die bestuur van die kliënte se verwagtings, die beskikbaarheid van die gebruikers en die teenkanting teen verandering. Die omgewing verwante uitdagings sluit die volgende in: die omgewing wat gedurig verander, die integrasie met 3de party stelsels, die taak om die verandering te bestuur asook die feit dat die kliënt-organisasie se kultuur nog nie gereed is vir verandering nie.

Tydens die studie is daar bevind dat al die projekte stelselsontwikkelingsmetodologieë gebruik. Die projekte wat betrokke was by die ontwikkeling en installering van ondernemings-hulpbron-beplanning (ERP) pakkette het almal die ASAP metodologie gevolg. Die ander projekte het gebruik gemaak van eie ontwikkelde metodologieë, waar

hulle dele gebruik het van die objek-georiënteerde asook die XP metodologieë. Geen van die metodologieë word streng gebruik nie. Dit word aangepas en op so 'n manier gebruik dat dit die verandering en uitdagings kan hanteer.

Die ondersteuning wat deur die stelselsontwikkelingsmetodologieë verleen word, sluit ook die volgende in: kontrole ondersteuning, kognitiewe asook koöperatiewe ondersteuning en produksie ondersteuning.

Die ondersteuning as kontrole tegnologie dui op ondersteuning van die bestuur van die projek, maniere om vordering te meet en om op datum en binne die begroting te bly. Hierdie studie het bevind dat die stelselsontwikkelingsmetodologie beide struktuur en rigting gee aan die ontwikkelingsproses en ook aanspreeklikheid verbeter. Die kognitiewe en koöperatiewe ondersteuning sluit die manier waarop die metodologie die span se samewerking bevorder in, asook die invloed wat dit het op kommunikasie en dokumentasie. Tydens die studie is daar gevind dat die gebruik van 'n stelselsontwikkelingsmetodologie die kommunikasie en dokumentasie verbeter. Dit verbeter die kommunikasie tussen spanlede asook die kommunikasie tussen die sagtewarehuis en die kliënt-organisasie. Ondersteuning as produksie tegnologie verwys na die gebruik van tegnieke en metodes wat help met die ontwikkelings proses. Die algemene gevoel by die sagteware huise was dat die bepaling van gebruikers behoeftes 'n moeilike fase is tydens die ontwikkelingsproses. Die studie het bevind dat verskeie gevalle metodologie gebaseerde tegnieke gebruik het om hierdie fase vir hulle te verbeter en vergemaklik.

Die projekspan lede se persepsies rondom die gebruik van stelselsontwikkelingsmetodologieë, het daarop gedui dat die gebruik daarvan 'n unieke verbetering inhou, veral met kommunikasie, dokumentasie en aanspreeklikheid. Hierdie is baie belangrike voordele vir sagtewarehuise, waar kliënte verhoudings en die bestuur van kennis noodsaaklik is.

CONTENTS

Chapter 1
Introduction

1.1 Introduction

Since the late 1960s the quality of developed systems and productivity of the systems development process have continued to be problematic. Faulty software was delivered late and exceeded the budget. In 1968, at a conference held to address this problem, the participants came to the conclusion that software engineering should use philosophies and paradigms similar to those used in other engineering disciplines. According to Shapiro (1997), this marked the beginning of systems development methodologies.

Various arguments exist for the implementation of systems development methodologies and also against the implementation of systems development methodologies. There is a widespread belief that devotion to systems development methodologies is beneficial to an organisation, and that using it can solve the software crisis (see Fitzgerald (1996) for a summary). Many developers recognise the advantages associated with the use of systems development methodologies (Hardy *et al.*, 1995; Avison & Fitzgerald, 2006). These advantages are considered to be both practical and intellectual (Jayaratna, 1994). Furthermore, it is generally assumed that systems development methodologies are used in practice. Saeki (1998) remarks that software development methods, also called methodologies, have been widely embraced and practiced because of the quality of the systems achieved using such systematic approaches.

Not all organisations favour the use of methodologies. Critics say that one could hardly apply the same methodology to different projects, since projects have more differences than similarities in common (De Marco, 1982). Another disadvantage that is mentioned by Avison and Fitzgerald (2006), is the amount of paperwork associated with methodologies. The failure to deliver the suggested productivity benefits is also a very common concern amongst critics.

The deployment of systems development methodologies is constantly influenced by these positive and negative arguments and therefore they are an important aspect of this study.

1.2 Research questions

This research will study the deployment of systems development methodologies at project level in software houses in South Africa. It will extend previous research of Huisman and Iivari (2002a, 2002b) who have studied the deployment of systems development methodologies at organisational level and at individual level.

Based on the work by Huisman and Iivari (2006), the author uses the term deployment to include the following three aspects namely the use, support and impact of the system development methodology.

Through this research the author will attempt to understand and explain:

- The context and the outsourcing environment in software houses where system development and the possible use of systems development methodologies take place.
- The use of systems development methodologies at project level in software houses in South Africa.
- The perceived support that the systems development methodologies provide. The term support includes production support, support in control and also cognitive and cooperative support.
- The perceived impact of systems development methodologies on the developed system and the development process.

The deployment of systems development methodologies can be studied from several perspectives. As can be seen in fig 1.1 there are three adopting units of system development methodologies. The highest level is the organisational level, in the middle is the project level and the lowest level is the individual level. These units are all intertwined. System development methodologies are seen as contingent innovations, which means that the methodology must first be adopted at organisational level before it can be adopted at

project level (Rogers, 1995). Therefore the one adopting unit has an influence on the other and in the end on the deployment process.

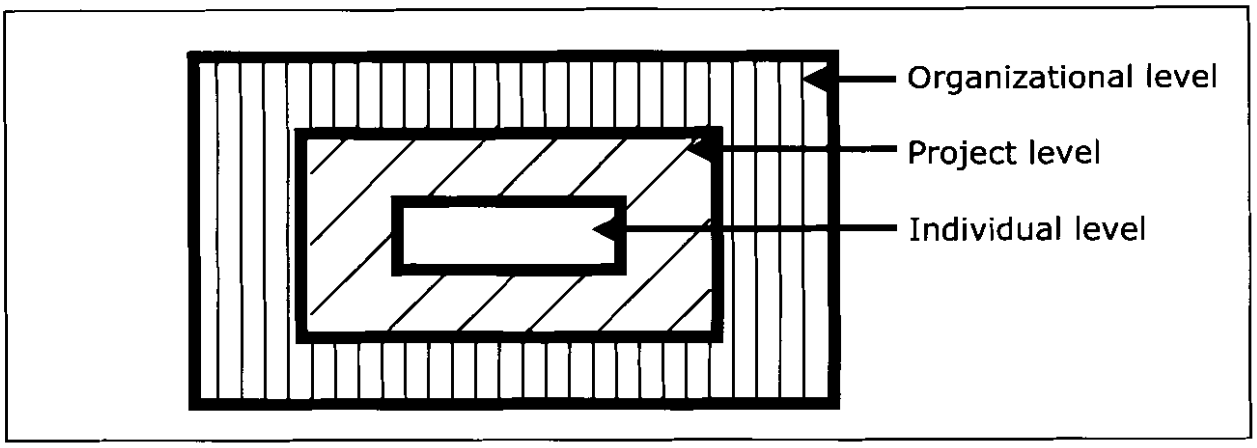

Fig 1.1 Adopting units of system development methodologies

Huisman and Iivari (2002a, 2002b) have studied the deployment of systems development methodologies at two of the levels, namely: the organisational level and the individual level.

This study will focus on the project level, thereby studying the project as the adopting unit in the deployment of system development methodologies in software houses in South Africa. A literature study has concluded that there has been limited research on the project as the adopting unit of system development methodologies.

According to De Villiers (2002), many issues including the environment need to be considered before choosing or introducing a methodology into an organization. In this study the author would like to study the environment in which the software houses operate, which includes the project team members perceptions on the outsourcing of information system development. Another part of the study is to examine the use, if any, of systems development methodologies at the software houses in South Africa. The selection of a specific methodology is normally done by the organisation (De Villiers, 2002). This illustrates that the project level is influenced by both the higher organizational level and the lower individual level and added to those effects; the project level also has its own issues to adhere to.

The importance of this research can not be doubted by either the research community or the practitioners. In the research community, this study could contribute to our knowledge of the deployment of system development methodologies in practice, and especially in the Software Houses of South Africa where system development is their core business. Also, this research could assist practitioners when they have to implement and use a systems development methodology. Lastly, academics and practitioners should work together, building on one another's skills and knowledge.

1.3 Reasons for the selection of this topic

- **Changing environment**

Fitzgerald (2000) mentions "the problem of tenses" where he states that the groundwork of most methodologies that are available today can be traced back to the 1970s. This is troublesome when taking into account how the technology and environment that development takes place in, has changed. The environment can also be described as increasingly turbulent with change being the norm rather than the exception (Hamel & Parahalad,1994). Successful organisations are often thought to be those that are capable of dealing with such change and the opportunities it presents.

In today's business environment speed and accuracy play a very important role. Information systems need to be developed faster to keep up with the constantly changing society. Through the internet and all the available technologies companies are globalising and the systems that support these companies need to be available and reliable.

Because of the constant changes in the development environment, the system development methodologies also need to change and adapt. Thus through this research the researcher will aim to explain how the South African software houses are using methodologies.

- **Outsourcing**

Outsourcing in relation to information technology is defined by Willcocks and Fitzgerald (1994) as the commissioning of a third party (or a number of third parties) to manage a

client organisation's information technology assets, people, and/or activities to obtain required results.

Multiple studies have analyzed the reasons that most often lead to information systems outsourcing (Gonzalez, *et al.*, 2005; Bahli & Rivard, 2004; Weidenbaum, 2005; Udo, 2000). It is clear that economic considerations are important when companies consider outsourcing. By drastically reducing information systems department size, companies can transform fixed costs into variable ones and, if the contract has been properly managed, into foreseeable costs (Alner, 2001). Then the client organisation knows what it will cost him to receive those services (Gupta & Gupta, 1992) and, in some cases, the client organisation may pass cost excess on to the software house. On the other hand, the client firm and the providing firm can take advantage of the economics of scale and scope obtained by using their specialized skills on a larger scale (Grover, *et al.*, 1994). It is easier and faster for the service provider (software house) to achieve a return on equipment-related costs, as the software house delivers services to many clients, thus achieving economics of scale.

However, there are not only economic reasons behind outsourcing. There is a whole group of strategic or business factors that drive firms to information systems outsourcing. Outsourcing makes it easier for firms to focus on their basic competencies (Grover, *et al.*, 1996). In this way, outsourcing liberates line executives, who do not have to coordinate themselves with a large information systems department, and the organization is simplified too. On the other hand, according to Alner (2001), there are also advantages related to information systems staff. The client has access to the knowledge of high-level specialists who do not have to belong to his own staff.

With the definition used above, the focus of outsourcing is on the result or outcome of the service being outsourced rather than on specifying how it is to be undertaken. This is a very important point, because this study will be looking at the way the service is provided, the methodologies used, and how the outcome might be influenced by these different factors. Unfortunately not all outsourcing has been successful and there is evidence of some degree of dissatisfaction (Avison & Fitzgerald, 2003). Organisations have terminated and renegotiated agreements. It is currently believed that there is more knowledge

available to software houses about the best areas to outsource and the processes to be undertaken to successfully deliver the desired outcomes. (Avison & Fitzgerald, 2002).

The outsourcing of information systems development has been an important subset of information technology outsourcing. The development of information systems is the core business of software houses, and these software houses may or may not use certain methodologies to develop the systems for their clients. Although this is an important subject, not much of literature regarding this topic was found. During this study the author will also try to determine whether the software houses in South-Africa use system development methodologies or not.

The client company outsourcing its system development has to develop skills in selecting the correct software house, specifying requirements in detail, and writing and negotiating contracts, rather than thinking about systems development methodologies (Avison and Fitzgerald, 2003).

With information systems outsourcing there are other potential, long-term effects for the client organisation as stated by Avison and Fitsgerald (2003):

> "The experience and expertise of developing and running systems in-house is being lost. The skills and expertise is being transferred to the vendors with the result that the organisation is increasingly dependent on outside vendors. This can be a big problem, if information technology is strategic to the business, or where it becomes strategic after outsourcing."

Further advantages and disadvantages of outsourcing will be discussed in chapter 2.

- **System development methodologies**

In the research done by Avison and Fizgerald (2006) on the different eras in system development, it can be seen how the environment is constantly changing. They identify the following eras regarding system development methodologies:

Pre-Methodology Era: Systems in the 1960s and 1970s were developed and implemented without formalized methodologies (Avison & Fitzgerald, 2006).

Early Methodology Era: Computer-based development shifted the focused to the identification of phases and stages that might help improve the management of systems development and introduce discipline. Research by Avison and Fitzgerald (2006) shows that this happened in the late 1970s and early 1980. This approach has become known as the Systems Development Lifecycle / Waterfall Model. Unfortunately shortfalls were not uncommon, systems failed to meet business needs, systems where unstable and inflexible. There also was an application backlog and documentation posed a problem.

Methodology Era: Approaches emerged in response to the above mentioned shortfalls. These approaches consisted of a recommended collection of phases, procedures, rules, techniques, tools, documentation, management, and training used to develop a system. Avison and Fitzgerald (2006) concluded that methodologies were adopted by organisations mainly because they wanted to achieve better end products, a better development process and a standardized process.

Era of reassessment: According to Avison and Fitzgerald (2006), researchers in the late 1990s have started questioning the concepts and usefulness of earlier methodologies, because the success or failure of development efforts cannot be attributed exclusively to the use, misuse, or non-use of methodologies. As a result, some organisations are still looking for newer (better) methodologies and approaches, while others have abandoned methodologies altogether. Avison and Fitzgerald (2006) see organisations moving in the following directions: Ad hoc development, which indicates the development of systems without using any formalized processes. External development, where Avison and Fitzgerald (2006) predict a movement to external development in a number of ways, including the use of packages and outsourcing. They see a contingency approach to information system development as another movement in this era. This entails the use of different approaches for different situations. Finally, according to Avison and Fitzgerald (2006), new methodologies will continue to be developed and the existing ones will evolve.

This study will particularly focus on the external development and the use of systems development methodologies that it entails.

1.4 Research Approach

Research into the deployment of systems development at project level in the software houses of South Africa can be based on a large number of research approaches, depending on the following:

- Research target.
  The major interest of this study lies in systems development methodologies as a contingent innovation deployed at project level in software houses in South Africa. The term deployment includes the use, support and impact of these systems development methodologies. The focus will be on systems development methodologies in general, and not on specific types of methodologies.

- Adopting units and stakeholder groups.
  The adopting units studied in previous research into systems development methodologies have varied. In this study the focus is on the project as the adopting unit of systems development methodologies. The study will also capture the perceptions of the project team members in the software houses in South Africa regarding outsourcing and the changing environment in which they operate.

- Research method.
  This study will use a qualitative research method to study the deployment of system development methodologies at project level in the software houses of South Africa. Multiple case studies will be done, whereby the data will be collected using semi-structured interviews. Content analysis will be done using cross-case analysis; the content analysis will be done with the help of ATLAS/ti software. Finally, member checking will be done to confirm the findings of the study.

1.5 Outline of the study

Chapter 1: Research problem
In this chapter the author defines the research problem and the research questions of the study.

Chapter 2: Outsourcing

The definition of outsourcing is stated, followed by the various arguments for and against outsourcing. The South African software houses, on which this study focuses, are the outsourcing service providers and are therefore directly involved in the whole process of outsourcing. A discussion on the primary concerns surrounding outsourcing is given. Following the concerns is the way outsourcing is perceived today and the factors that influence the decision to outsource.

Chapter 3: Systems development methodologies

In this chapter the definition of a systems development methodology as well as an explanation of the deployment of such a methodology is explained. The definition is followed by arguments for and against systems development methodologies, after which the role of systems development methodologies is discussed. Lastly, previous research on systems development methodologies will be discussed with special attention to literature where the focus is on the deployment of system development methodologies at project level.

Chapter 4: Research Method

In this chapter the author discusses the nature of qualitative research. This will then be followed by a discussion of the qualitative research method used in this study.

Chapter 5: Results

In this chapter the author will present the findings of the study. The author will start by re-stating the aims and objectives of the study, after which he will describe each case that was used in this study. Thirdly the environment related findings will be discussed, after which the findings with regard to the deployment of systems development methodologies will be discussed.

Chapter 6: Discussion and Conclusion

In the final chapter the author presents a summary of the results, and discusses shortcoming of this study as well as possible future research.

Chapter 2
Outsourcing

2.1 Introduction

In this chapter previous research on outsourcing is reported. The South African software houses, on which this study focuses, are the outsourcing service providers and are therefore directly involved in the whole process of outsourcing in South Africa. The definition of outsourcing is stated, followed by the various arguments for and against outsourcing. A discussion on the primary concerns surrounding outsourcing is given. Following the concerns is the way outsourcing is perceived today and the factors that influence the decision to outsource. The chapter is concluded with a summary.

2.2 Definition of Outsourcing

Outsourcing in relation to information systems is defined by Willcocks and Fitzgerald (1994) as the commissioning of a third party (or a number of third parties) to manage a client organisation's information systems' assets, people, and/or activities. Another definition of information system outsourcing is that by Kishore *et al.* (2003) as the contracting of various information system functions such as managing of data centres, hardware support, operations, software maintenance, network, and even application development to outside service providers.

According to the above-mentioned definition by Wilkocks and Fitzgerald (1994), the focus of outsourcing from the clients' point of view is on the outcome of the service being outsourced and not on how it should be done. However, in this study the focus will be on the way the service is provided and the accompanying methodologies used, if any.

Another type of outsourcing, also called offshoring, has become increasingly popular (Avison & Fitzgerald, 2006). Offshoring involves the outsourcing of services to offshore service providers. Financial benefits in terms of labour costs seem to be the main driving force of offshoring (Avison & Fitzgerald, 2006).

According to Chaudhury and Subherwal (2003), Lacity and Wilcocks (1998) and Kishore *et al.* (2003) it is believed that outsourcing has an important strategic impact and that an organisation's information technology and systems portfolios can be managed effectively by an outsourcing service provider. An important point to remember is that outsourcing relationships are not static; they tend to change and evolve over time due to changes in the external environment and in clients' internal requirements (Weakland, 2005). Earlier research by Kishore *et al.* (2003) developed a framework that classifies client-provider outsourcing relationships into Four Outsourcing Relationship Types. This is called the FORT Framework, and is presented in fig 2.1.

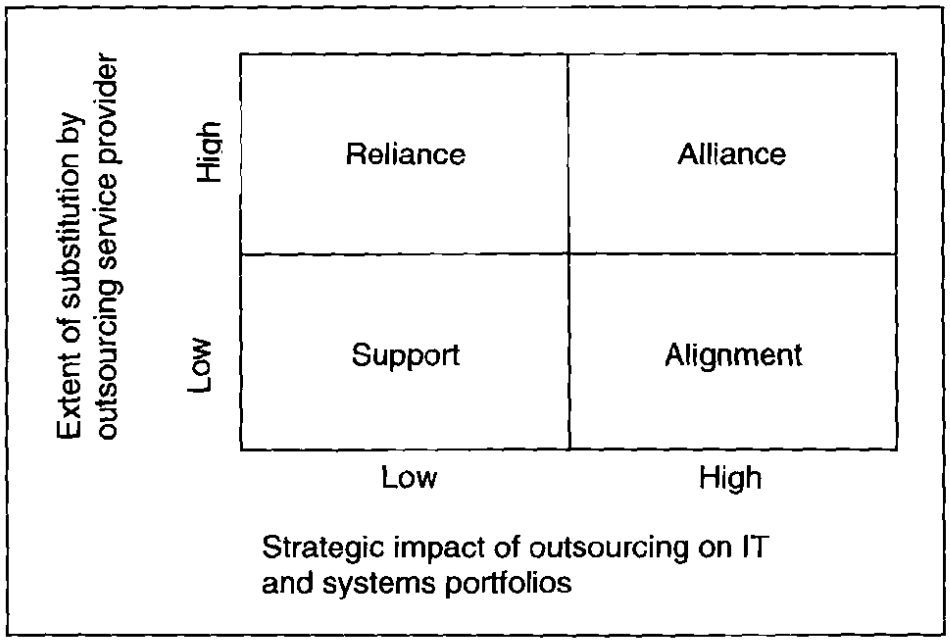

Fig 2.1 Fort Framework. Kishore *et al.* (2003)

The FORT framework consists of two dimensions. The first dimension deals with the extent of ownership substitution by outsourcing service providers. Ownership substitution focuses on the degree to which ownership and/or control of various information system and technology assets have been transferred to the software house. The second dimension deals with the strategic impact of outsourced information system and technology portfolio.

This dimension captures the influence the outsourced portfolio or system has on a firm's competitive positioning and its long-term strategy.

A client firm's relationship with an outsourcing service provider firm can be classified into one of the following four generic types of client-provider relationships:

Support

The support relationship involves low coordination costs, and monitoring the relationship is easy because the extent of substitution by the vendors is low. This is typically the traditional information system services such as payroll processing. The firm in this quadrant would usually use in house development instead of outsourcing. They would only outsource on a selective basis to support the information services of the firm.

Alignment

In the alignment and alliance relationships, coordination is much more complex and monitoring becomes more difficult. In an alignment relationship the extent of substitution is low and the strategic impact is higher. The services are generally consulting type high impact information systems services, but they are mostly project based such as the implementation of a packaged solution. The main motivator in this quadrant is the access to world class technical knowledge and expertise.

Alliance

Alliance relationships typically grow and build upon previous small, but successful, exchanges between organizations. Both the extent of substitution and the strategic impact are high and it is seen as the most comprehensive type of outsourcing relationship. The relationship involves the management of a strategic type of partnership with the service provider.

Reliance

This type of relationship is normally outcome based with a low strategic impact and high substitution. Cost reduction is typically the major motivator for this type of outsourcing relationship, and the contract periods are usually longer term.

The above-mentioned types of outsourcing relationships also require different competencies and monitoring mechanisms. For example, the support relationship involves low coordination costs and monitoring the relationship is easy because the extent of substitution is low. By contrast, coordination in the alliance and alignment relationships is much more complex and monitoring accordingly becomes more difficult. This is because specifications for outsourced services are difficult to specify completely, and outcomes could be ambiguous and uncertain. In the case of an alliance type of relationship, trust rather than incentives and penalties becomes an important mechanism to ensure that service providers' interests coincide with clients' interests (Kishore *et al.* 2003).

2.3 Arguments for Outsourcing

The reasons that most often lead to information system outsourcing have been analysed by numerous researchers. The following reasons for outsourcing are given:

- To obtain IT services at lower cost

Outsourcing can transform fixed costs into variable ones and, if the contract has been properly managed, into foreseeable costs (Alner, 2001). The client knows what it will cost to receive those services (Gupta & Gupta, 1992) and, in some cases, he may pass cost excess on to the service provider. On the other hand, the client firm and the providing firm can take advantage of the economics of scale (Grover *et al.*, 1994). It is easier and faster for the providing firm to achieve a return on equipment-related costs, as the firm delivers services to many clients (Gonzalez *et al.*, 2005).

- Tax advantage

The tax advantage comes from the ability to deduct the outsourcing expenses from the current year's earnings (Alner, 2001).

Cash flow improvements, software licensing costs and personnel costs are transferred to the service provider (Alner, 2001; Gonzalez *et al.*, 2005).

- Retaining top personnel

This becomes the responsibility of the service provider. He has to keep his personnel up to date and handle any turnover that may occur (Alner, 2001; Gonzalez *et al.*, 2005).

- Gain access to leading-edge technologies

An external service provider enables an organisation not only to utilize the service provider's technologies but also to tap into his links with other technology providers and users (Akomode *et al.*, 1998; Gonzalez *et al.*, 2005).

- Flexibility

Given the increasingly rapid pace of technological changes witnessed in recent years, many organisations gain a significant advantage from information system outsourcing, especially in the sense that outsourcing service providers will not become technologically obsolete. Client firms can increase their flexibility by permanently redesigning their contracts to meet their information needs (Gonzalez, 2005).

Although economic reasons are important, they are not the only reasons behind outsourcing. There is a whole group of strategic or business factors that drives firms to information systems outsourcing. Outsourcing makes it easier for firms to focus on their basic competencies (Grover *et al.*, 1996) and in this way, outsourcing liberates line executives, who do not have to coordinate themselves with a large information system department. On the other hand, there also are advantages related to information systems staff (Lacity *et al.*, 1996). The client has access to the knowledge of high-level specialists who do not have to belong to his own staff (Alner, 2001). Another advantage of outsourcing is that several service providers can be used simultaneously by a client firm depending upon the service providers' expertise and special offerings. In addition it allows the client firm to minimize costs and maximize business benefits simultaneously (Gonzalez *et al.*, 2005).

2.4 Arguments against Outsourcing

The following is a discussion on arguments against outsourcing that was found in literature.

Not all outsourcing has been successful and there has been some degree of dissatisfaction as stated by Avison and Fitzgerald (2003) and Earl (1996). Organizations have terminated and renegotiated agreements, which left client companies frustrated and disappointed.

Potential long-term effects for the client organization are also threatening (Avison & Fitzgerald, 2003). Excessive dependence on the supplier as discussed earlier can become a big problem when the quality of the outcomes provided by the software house is not up to standard (King & Malhotra, 2000). The client organization then loses the skills, experience and expertise that they would have had if they had developed the systems in-house. The skills and expertise are being transferred to the software house and the organization is therefore increasingly dependent on the software house. This becomes a major hurdle when information technology is strategic to the organization's business (King & Malhotra, 2000).

Organisations enter outsourcing agreements with the objective of cutting costs and improving the level of service rendered to users. According to Alner (2001) and Gonzalez *et al.* (2005), outsourcing is reported to be financially beneficial to an organisation, but there are costs that need to be considered. Therefore the outcome of such contracts may have just the opposite effect. There are several costs that client organisations neglect to consider, such as unexpected transaction and management costs (Barthelemy, 2001; Dué, 1992). These costs can represent an important proportion of the total costs of an outsourcing agreement (Scheier, 1996). Sometimes these additional costs represent the very amount that a firm was expecting to save by entering into an outsourcing agreement. Examples have also been provided where the actual costs of services rendered by the supplier were much higher than those related to internal provisions of the same services (Lacity & Herscheim 1993).

Although the advantage of being able to select more than one service provider is obvious (Lacity & Wilckocks 1998), it can create coordination problems among multiple service providers, and client firms should pursue this strategy only if they are capable of coordinating activities of multiple service providers simultaneously.

The irreversibility of the outsourcing decision, once it has been made, also tends to keep sceptics at a distance . Referring to 2.1 in the reliance quadrant, the contract periods tend to be long and the switching to new technologies can be a difficult process.

Disagreements between the parties are also reported (Earl, 1996). Such disagreements may lead to disputes and serious problems, which are not only disruptive, but can also be very costly. Apart from the direct costs of lawyers and experts' fees, and the costs of the in-house resources whose time is spent working on the litigation, indirect expenses, associated with reputation effects, may also be incurred.

2.5 Primary Concerns regarding outsourcing

Outsourcing concerns are a harsh reality that is haunting management around the world. Weakland (2005) compares the outsourcing concerns perceived by the client organisations versus the outsourcing concerns perceived by the software houses. In Fig 2.2 the perceived concerns are listed on the left and the occurrence as perceived by the client organisations (black) and the software providers (grey) are indicated by the horizontal bars.

These concerns include the following:

- Increased management complexity.
- Reduced effectiveness due to communication difficulties.
- Lower quality of output.
- Lack of direct control over resources.
- Uncertain financial payback.
- Uncertain information confidentiality.
- Lack of proximity to staff.

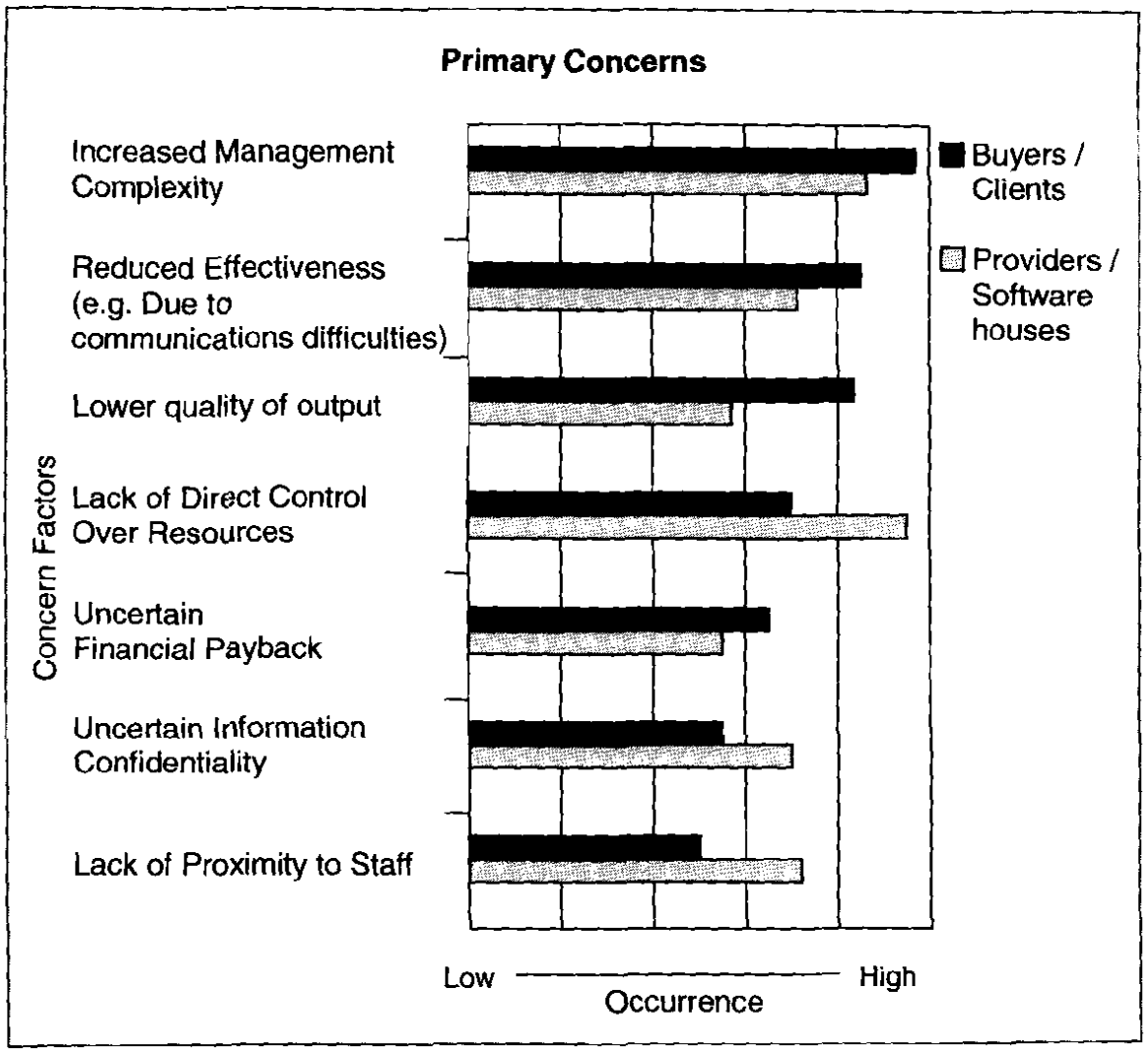

Fig 2.2 Primary Concerns. Weakland (2005:7) DiamandCluster 2005 Global IT Outsourcing study

According to fig 2.2, increased management complexity was found to be the greatest concern to client organisations. It was followed closely by reduced effectiveness and lower output quality. Software houses saw the lack of direct control over the resources as the biggest problem. It was followed by the increased management complexity and in an almost combined 3rd place was the lack of proximity to staff, uncertain information confidentiality, and reduced effectiveness. As can also be seen in fig 2.2, there is a big gap in the perceptions between software houses and client organisation regarding the perceived concerns of outsourcing.

2.6 Outsourcing today

According to Lemmon (2005), the demand for outsourced information system services is rapidly rising, as more and more companies look for ways to cut their information system costs and improve productivity. Lemmon (2005) states that global spending on outsourcing will increase tremendously; however, not all of the projects will succeed.

White (2003) states that outsourcing has grown at a phenomenal rate throughout the world, including in South Africa. The many changes in employer-employee management structures in South Africa, since 1994, provided the platform for the growth of outsourcing as an industry, and many sophisticated outsourcing brands now provide professional and reliable services. Outsourcing suppliers are able to supply the emerging companies of South Africa and around the world with the backing of educated and experienced people, which allows their attention to be on their core business, and to supply their focused services to their client.

Other reasons for the growth in South Africa's outsourcing industry include the 2010 FIFA world cup that is stimulating the development of information and communication systems and the fact that South Africa's telecommunications infrastructure is the most sophisticated in Africa (Austrade, 2006).

According to Austrade (2006), the South African IT sector is very competitive and has almost too many competitors in certain sectors. As a result, many software houses are expected to reassess business strategies to reflect specialisation and to focus on niche markets.

2.7 Factors that influence the decision to outsource

Companies planning on outsourcing some information system functions should do so ahead of time, carefully considering a broad range of issues in order to ensure that these projects will be successful. Lemmon (2005) outlines common pitfalls for companies that outsource information system services. These include the following factors that companies

often fail to fully consider when deciding whether or not to outsource information system functions:

- Cost

Travel, training and communications costs are often overlooked (Scheier, 1996). Companies should also consider the costs of the different stages of outsourced projects. Generally initial costs are higher and the organization will only realize cost savings after significant time has passed (Lemmon, 2005).

- Productivity

According to Lemmon (2005), the productivity levels may vary throughout the lifecycle of the outsourced project. This is affected by, including others, staff turnover at the outsourcing provider / software house.

- Communications

Effective communication is vital, as discussed by Lemmon (2005) and Scheier (1996). Documenting all communications between the organization and the service provider can help to avoid misunderstandings. This is especially important when the technical requirements and business objectives are discussed.

- Culture

Cultural differences may affect the working relationship between the organization and the service provider. A possible solution, mentioned by Lemmon (2005), is to seek advice from consultants and consider cultural training.

- Organizational readiness

The organization's maturity can also play a vital role in the success of an outsourcing relationship as mentioned in Lemmon (2005). Important factors for both parties to consider are standardized methods of relationship management, project management skills and service level agreements.

- Offshore vs. onshore service providers

Also influencing client organisations when choosing service providers is the offshoring possibilities that exist. Weakland (2005) reveals that offshoring clients in growing numbers are: dissatisfied with offshore service providers, prematurely terminating contracts and struggling to harvest the full value of their outsourcing relationships. While India and the US are still the top locations for off-shore services, interest in China is growing, which is bound to put downward pressure on rates (Weakland, 2005). But while this evidence of a maturing market suggests challenges for outsourcing and offshoring providers, the news appears better for the information systems staffs at US companies. While senior executives still view outsourcing as a cost-cutting opportunity, they also recognize outsourcing's value as a means to manage variable demands from the rest of the business and to redeploy in-house information systems personnel for more crucial purposes.

When considering buyer satisfaction, the picture for offshore providers is not very pretty. Weakland (2005), states that a significant change is that buyers no longer prefer the services of their offshoring providers over that of their onshore providers. According to Weakland (2005), 74 percent of study participants were satisfied with their onshore outsourcing initiatives in 2004. In 2005 that number rose to 81 percent. Offshoring satisfaction rates, however, have fallen significantly over this period, from 79 percent to only 62 percent today. The factors that are believed to contribute to the decline in offshoring satisfaction include, among others, competition and complexity.

The fierce competition for the best resources has led to unexpectedly high employee turnover rates, making it difficult for many offshore outsourcing firms to keep their staffing commitments to their buyers. The huge demand for offshore resources has also caused tremendous growth in the number of firms providing outsourcing services. While most of these firms are of high quality, it is inevitable that some are not. As differentiating between offshore providers remains difficult, buyers risk selecting lesser quality firms based solely on price differential (Weakland, 2005).

As offshore firms continue expanding their breadth of capabilities, the work becomes more complex and less commoditised. As a result, many firms are learning about industry and business processes while busy with the strategically vital processes of the client. This

leads to higher costs, more missed deadlines and overall lower client satisfaction. Despite the downturn in satisfaction with offshoring over the past year, offshoring still remains convincing. As the industry matures, however, buyers need to become more rigorous in thinking about their outsourcing strategies and utilize the available tools and processes to confirm that they are using the right resources for the right reasons (Weakland, 2005).

2.8 Ranking of selection criteria

Weakland (2005) shows that the organisation's ranking of important factors that influence the decision to outsource and also the choice in service provider have changed through the years (fig 2.3).

Client Organisation's Ranking of Key Selection Criteria		
2005	2004	2002
1. Technology expertise	1. Reference and reputation	1. Cost
2. Cost	2. Cost	2. Resource skill set breadth
3. Flexibility in structuring operating model	3. Resource skill set breadth	3. Size of vendor
4. Existing or prior relationship	4. Industry expertise	4. Flexibility of contract
5. References and reputation	5. Existing or prior relationship	5. Process quality

Table 2.1 Key Selection Criteria. Weakland (2005:14) DiamandCluster 2005 Global IT Outsourcing study

Software houses must be willing to partner with client organizations in a wide variety of models and must show a willingness to adapt over time as dictated by changing needs and business conditions (Weakland, 2005). Client organizations are more aware of the important factors one should consider when choosing a software house and, according to Weakland (2005) (table 2.1), organisations have shifted their criteria to a ranking where

expertise, cost, flexibility, relationships and references are the top factors to consider when selecting a service provider.

2.9 Summary

The term outsourcing as well as the relationship between the client organization and the supplier (software house) was defined in this chapter. Arguments for and against outsourcing were mentioned, followed by a discussion on the primary concerns surrounding outsourcing. Then the current situation of outsourcing was reported, followed by factors that influence the decision to outsource.

Chapter 3
Systems Development Methodologies

3.1 Introduction

In this chapter the definition of a systems development methodology as well as an explanation of the deployment of such a methodology is explained. As no universally accepted definition for a systems development methodology exists, such a discussion is of vital importance. The definition is followed by arguments for and against systems development methodologies, after which the role of systems development methodologies is discussed. Lastly, previous research on systems development methodologies will be discussed with special attention to literature where the focus is on the deployment of system development methodologies at project level.

3.2 Defining a systems development methodology

Defining a systems development methodology is the first and probably most important step to be taken before studying the deployment of systems development methodologies at project level. According to Avison and Fitzgerald (2006), the term methodology is not well defined in the literature or by practitioners. Many people prefer the term method over methodology and therefore there is very little agreement as to what is meant by a methodology. The term is used very loosely in literature and in practice as to incorporate many of the available definitions. The following is a definition given by Avison and Fitzgerald (2006):

> "A system development methodology is a recommended means to achieve the development, or part of the development, of information systems based in a set of rationales and an underlying philosophy that supports, justifies and makes coherent such a recommendation for a particular context. The recommended means usually includes the identification of phases, procedures, tasks, rules, techniques, guidelines, document management and organization of the approach and the identification and training of the participants."

Another definition is that by Fitzgerald *et al.* (2003) where they use the word method instead of methodology. They define a method as:

> "A coherent and systematic approach, based on a particular philosophy of systems development, which will guide developers on what steps to take, how these steps should be performed and why these steps are important in the development of an information system."

In the study by Huisman (2000), a summary of definitions was presented and from this summary she stated that a systems development methodology is a combination of the following:

- A systems development approach/approaches:
  The philosophical view on which the methodology is built. It entails the set of goals, guiding principles and beliefs, fundamental concepts and principles of the systems development process that drive interpretations and actions in the development of systems. Examples are the structured approach, object oriented approach, information modelling, etc.
- A systems development process model/models:
  A representation of the sequences of stages through which a system evolves. Some examples are the linear life cycle model and the spiral model.
- A systems development method/methods:
  A systematic approach to conducting at least one complete phase of systems development, consisting of a set of guidelines, activities, techniques and tools, based on a particular philosophy of systems development and the target system. Examples include OMT, IE, etc.
- A systems development technique/techniques:
  A procedure, possibly with a prescribed notation, to perform a development activity, for example entity relationship diagrams.

In this study the term methodology, as explained above by Huisman (2000), will be used.

3.3 Arguments for systems development methodologies

In the system development community there are many arguments for and against the use of systems development methodologies. The deployment of systems development methodologies is constantly influenced by these positive and negative arguments and therefore they are an important aspect of this study. A discussion of these arguments follows.

- Providing a standard

The first and probably best known advantage of systems development methodologies is the standardisation of design development and implementation procedures. This category relates to the benefits of having a common approach throughout an organisation. Standardisation of the development process facilitates interchangeability among developers and improves management and project control. This means that staff can change from project to project without retraining being necessary. It may also lead to increased productivity as resource requirements can be predicted and made available as and when necessary. Easier maintenance and enhancement is another benefit brought about by standardisation (Kruchten, 2001; Meso *et al.*, 2006; Fitzgerald, 1998; Avison & Fitzgerald, 2006).

- Ensuring product quality

Most methodologies ensure product quality by providing a framework of processes with measurement and criteria for their execution. The quality required for outputs or deliverables are specified by many methodologies. Some of these components that contribute to the quality of the output include acceptability; availability; cohesiveness; compatibility, documentation; economy; effectiveness; efficiency; fast development; maintainability and upgradeability; functionality; implementability; low coupling; portability; reliability; robustness; security; simplicity; testability; timeliness; visibility and simplicity (De Vries, 2004; Fitzgerald, 1998; Purvis & Sambamurthy, 1997; Avison & Fitzgerald, 2006).

- Control of the development process

Systematic activities specified by methodologies help organisations to keep track of system changes, especially when going through the iterative requirements, design and

implementation phases. By making the development task more visible and transparent methodologies may facilitate project management and control of the development process. It is also argued that the use of a methodology reduces the level of skills required of the analyst. By improving the development process, systems can be built faster and at a lower cost (De Vries, 2004; Fitzgerald, 1998; Avison & Fitzgerald, 2006).

- Ensuring re-usability

Certain methodologies are designed to support component-based development. Due to the implementation of the concepts of modularity and encapsulation, these components may be re-used in different information systems, reducing the overall development time of new systems. As mentioned above, certain components that are necessary to obtain reusability are incorporated into the methodologies (Avison & Fitzgerald, 2006).

- Learning

A methodology may provide a structural framework for the acquisition of knowledge, therefore past development experiences can be systematised and stored for future reference. Meso *et al.* (2006) explain that the developers working on a systems development project are consistently processing information as they generate a solution for the business problem they are addressing. By doing this, they obtain new insights into the business, the problem, the solution being developed and the processes that are guiding the development effort. These insights are obtained through the various processes of creation, transfer, sharing, storage, search, discovery and use of knowledge at both individual and team levels (Fitzgerald, 1998; Glasser, 1998).

3.4 Arguments against systems development methodologies

Not all organisations favour the use of methodologies. They reason that one could hardly apply the same methodology to different projects, since projects have more differences than similarities in common (De Marco, 1982). The amount of paperwork associated with methodologies can also result in a loss of motivation among developers. Following is a list of criticisms on system development methodologies found in the literature.

- Productivity

The failure to deliver the suggested productivity benefits is a very common concern amongst critics. It is said that the use of system development methodologies might increase the time it takes to develop an information system. This is normal because, when using a methodology, the development team has to do more tasks, construct more diagrams and models and do a lot of documentation (Avison & Fitzgerald, 2006).

- Inflexible

The methodology might not allow for requirement changes to be made during development. This is problematic as requirements frequently change during the long development process (Avison & Fitzgerald, 2006).

- Assumptions and generalisations

Exceptions are not catered for in systems development methodologies. As all developers know, projects might be similar but are seldom, if ever, exactly the same. Information systems projects have a tremendous amount of variables, including organisational and individual variables. The rationale and sequential processes of the methodology seldom fit all organisations (De Vries, 2004).

- Concentrate on technicalities

It is a common trend in development to regard information system development as a routine process where the main goal is obtaining the best suited technology to solve the problem. However, social aspects play an absolutely vital role in system development and need to be incorporated (De Vries, 2004; Avison & Fitzgerald, 2006).

- Unsuitable for rapid development

Fitzgerald (1998) indicates that the organisational environment has changed to such an extent that many methodologies are no longer useful. His study indicates that methodologies are used if five or more developers are employed and when the project duration exceeds nine months. This is troublesome as in today's business environment speed and accuracy play a very important role. Information systems need to be developed faster to keep up with the constantly changing society. Therefore, traditional approaches

resulting in the eventual delivery of systems only after several years, are no longer appropriate.

- System development is not an orderly rational process

System development is not a routine process; there are more differences than similarities between systems. Still most systems development methodologies are designed as if system development is a fixed process, almost a recipe, with major emphasis on technical rationality instead of or at the expense of social aspects (Fitzgerald, 1998).

- Difficulties in adopting a methodology

Organisations have found resistance from developers who are experienced in more informal approaches to system development. These developers often see the use of a methodology as restrictive and unnecessary (Avison & Fitzgerald, 2006).

- Complexity of a methodology

Systems development methodologies are designed to be applied to large and comprehensive development projects. They tend to specify every possible task that might be relevant. All of these tasks are then expected to be undertaken for every development project no matter what the size (Avison & Fitzgerald, 2006).

- "Gilding the lily"

The use of methodologies tends to cause extensive development of any requirements to a degree over and above that which is required. Every requirement is being treated as being of equal importance, which is seldom an accurate representation of the situation (Avison & Fitzgerald, 2006).

- Skills in using a methodology

Methodologies require a significant amount of skills in their use and processes. Methodology users and end users often have difficulty in learning the processes and in acquiring these skills needed. It is even argued that the use of system development methodologies does not improve system development skills or organisational learning (Avison & Fitzgerald, 2006).

- Tools

Many of the tools incorporated in the methodology are difficult to use and does not generate enough benefits. It is said that these tools increase the focus on the production of documentation instead of focussing on better analysis and design (Avison & Fitzgerald, 2006).

- Not contingent

The methodology is not always contingent upon the type of project or its size. Therefore the application of the whole methodology is done, irrespective of its relevance (Avison & Fitzgerald, 2006).

- One dimensional approach

Usually the methodology specifies only one approach to the development of projects. This causes the methodology to not always address the underlying issues and problems (Avison & Fitzgerald, 2006).

- Goal displacement

It is said that the use of a methodology can lead to a focus on following the process instead of solving the problem at hand and addressing the user's needs. This can result in developers working hard and diligently but avoiding the real problems of effectively developing the required system (Avison & Fitzgerald, 2006).

- Problem of building understanding into methods

It is argued that some methods assume that understanding can be built into the method process and that developers need little or no understanding of the problem situation, because the method will somehow bring to light all the characteristics that need to be discovered (Avison & Fitzgerald, 2006).

- No improvements

It is the conclusion of some that the use of methodologies has not resulted in better systems. It was found that users have found the methodology to actively hinder the success of the project (Avison & Fitzgerald, 2006).

3.5 The role of systems development methodologies

Software houses have to produce quality systems fast and within budget limits. The management of the available resources and time is extremely important. Information systems development methodologies are used by some organisations to structure the information systems development process. Many methodologies additionally include project management components to facilitate the management of the development process (De Vries, 2004). This study will examine whether the South African software houses use system development methodologies and the different roles these methodologies may play.

System development is an activity involving and affecting many different stakeholder groups. According to Fitzgerald *et al.* (2002), the roles of a methodology can be divided into two broad categories. These categories are rational and political.

Rational roles of systems development methodologies

- The complex information systems development process can be addressed by methodologies that subdivide the process into more plausible and coherent steps.
- Facilitation of project management and control makes the development task more visible and transparent, thus reducing risk and uncertainty.
- Methodologies may allow skill specialisation and a division of labour in the information system development process. Methodologies also have an economic role, for example money can be saved by breaking down the process into discreet activities that can receive different remuneration rates.
- Methodologies also play a role in providing a framework for the acquisition of knowledge. This is done by the systematisation of development knowledge whereby experience and knowledge, instead of being lost, is stored for future reference.
- Standardisation of the development process facilitates inter-changeability among developers. Methodologies can also lead to increased quality and productivity, as predictions of the resources needed can be done more accurately.

Political roles of systems development methodologies

- Professionalizing information system development by helping developers agree on reasonable deadlines and demands from departments.
- Enabling information system departments to be more proactive in strategy formulation.
- Providing comfort / confidence, whereby the use of methodology provides some reassurance that a certain process is being followed and decisions have been made on a systematic basis.
- The use of methodologies may also help in providing an audit trail of the development process. This offers protection in the future when information systems development decisions turn out to be wrong.
- Organisations may use methodologies to help achieve ISO-certification.
- Method champions can use it to raise their profile by providing the methodology as a power-base.

3.6 Historical perspective of system development methodologies

According to Avison and Fitzgerald (2006), the history of system development methodologies can be broken into the following eras:

- Pre methodology era

This represents the early 1970s, when system development was done without the use of any formalized methodologies. Developers where technically trained but seldom fully understood the organizational context in which the systems were being implemented. The user's needs were rarely well established and this approach often resulted in poor control and management of the project.

- Early methodology era

This era was characterized by an approach that focused on the use of phases and stages when developing an information system. This approach was called the system development life cycle or the waterfall approach. It was thought to help control and enable the better management of system development; unfortunately it also had weaknesses and

shortfalls. These include, amongst others, the failure to meet management needs, inflexibility, documentation problems and major problems with the naïve idea of an "ideal" situation and approach (Avison and Fitzgerald, 2006).

- Methodology era

In this era the use of methodologies was seen as a response and possible solution to the limitations and concerns of the previous era. The term methodology was used to describe the different approaches to system development. In the time from the mid 1980s to the late 1990s the use of a methodology of some kind, albeit in-house developed methodologies, which might or might not be based on a named commercial methodology, was very popular amongst organizations. In this era the use of methodologies expanded to include, amongst others, business and strategic planning phases and the integration of new technologies.

- Era of methodology reassessment

This represents the era that we are currently in. It is characterized by a serious reappraisal of the concepts and practicalities of the methodologies of the methodology era. In Avison and Fitzgerald (2006) it is mentioned that they see organizations moving in the following directions:

Ad hoc development: The development of systems without using any formalized processes. This approach relies heavily on the expertise and experience of the developers. It can almost be seen as a return to the approach followed in the pre methodology era.

Further development in the methodology arena: According to Avison and Fitzgerald (2006), new methodologies will continue to be developed and the existing ones will evolve. However, they believe it unlikely that a single methodology will provide the solution to all the system development problems.

Contingency: Current methodologies are designed for the situations that follow an 'ideal type'. However, all projects are different and that ideal type seldom if ever exists. Therefore Avison and Fitzgerald (2006) state that they see a contingency approach to information system development as the third movement in this era. This entails the use of different approaches for different situations, taking into account factors such as the type of project

and its objectives, the organization and its environment, and the users and developers and their respective skills.

External development: Avison and Fitzgerald (2006) also predict a movement to external development in a number of ways, including the use of packages and outsourcing. The package market is becoming increasingly sophisticated and client companies can buy a package that covers all their standard requirements. These packages can then be tailored to fit the client organization's needs and only the strategic systems, for which packages does not exist, are developed in-house. The enterprise resource packages have emerged in the last few years and have become very popular with larger corporations. The key for client companies is to ensure the correct trade off between a standard package, meaning they might have to change some of their business functions to fit the package, and a package that can be modified to adapt to the way they operate. There are dangers of becoming too dependent on a service provider. Avison and Fitzgerald (2006) state that system development problems and the perceived failure of system development methodologies have resulted in the growth in information system outsourcing. When outsourcing information system development, it is the effectiveness of the system delivered that is important to the client organization and not the development process. The outsourcing of system development efforts entail that the client company has to choose a service provider. This is a very important decision and skills in selecting this correct service provider is becoming a matter of necessity.

The focus of this research includes a study of the environment in which South African software houses operate, as well a study of their use of systems development methodologies. Many client organsations rely on software houses to develop high quality systems in budget and on time; this represents the external development as described by Avison and Fitzgerald (2006). Although clients are mainly concerned about the end product and not the process, this study investigates the process as it possibly has an ifluence on the quality of the outcome.

3.7 Deployment of systems development methodologies at project level

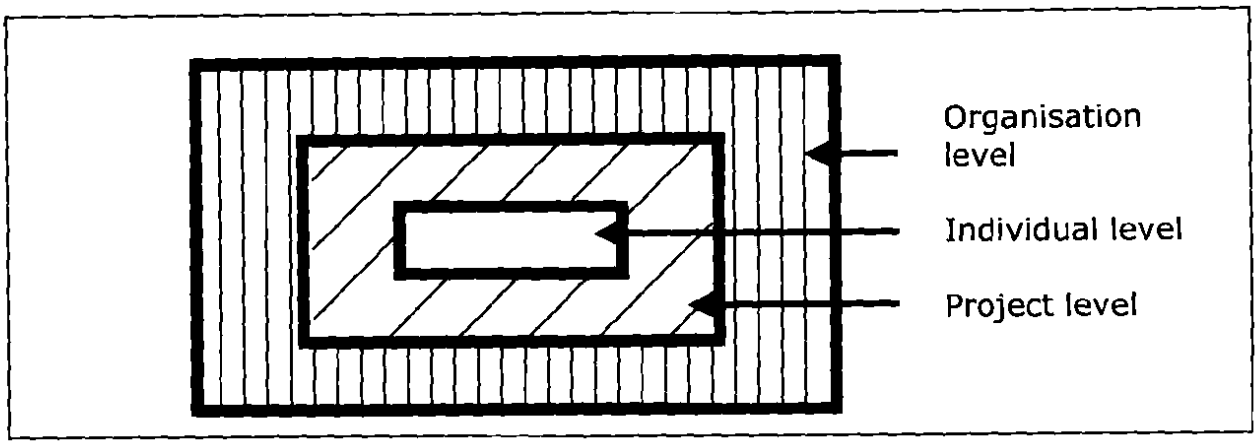

Fig 3.1 Adopting units

A system development methodology is seen as a contingent innovation (Rogers, 1995). First the organization needs to adopt the methodology, after which the methodology is adopted at project level and finally the individual developers adopt the methodology. At the organisational level the focus is on the characteristics of the organisation. This includes aspects like the organisational culture, maturity and the business area. Huisman and Iivari (2002a) studied the deployment of systems development methodologies at organisational level and found that relative advantage; compatibility; maturity of an information systems department; the perceptions of hierarchical culture and the amount of time an information systems department spends on the development of new systems all showed a positive relationship to the deployment of systems development methodologies at organisational level. They found that there is a negative relationship between the perceptions of rational culture and the organisational deployment of system development methodologies.

At the individual level the focus is on the characteristics of the individual. This includes aspects like the experience of the individual, his education and the impact of uncertainty on the use of a systems development methodology. Huisman and Iivari (2002b) studied the deployment of systems development methodologies at individual level and found that relative advantage; compatibility; management and developer support; individual's experience with system development methodologies and trialability all had a positive

relationship with the individual deployment of system development methodologies. Their results did not show any relationship between education and the individual deployment of system development methodologies. Furthermore a negative relationship between an individual's experience in system development methodologies and the individual deployment of system development methodologies was found.

This research will extend the previous research by Huisman and Iivari (2002a & 2002b) by focusing on the deployment of system development methodologies at project level. The term deployment is used in this study as stated by Huisman and Iivari (2006) to include the following three aspects, also shown in fig 3.2:

1. The use of systems development methodologies.
2. The perceived support that the systems development methodologies provide. The term *support* includes production support, support in control and also cognitive and cooperative support.
3. The perceived impact of systems development methodologies on the developed system and the development process.

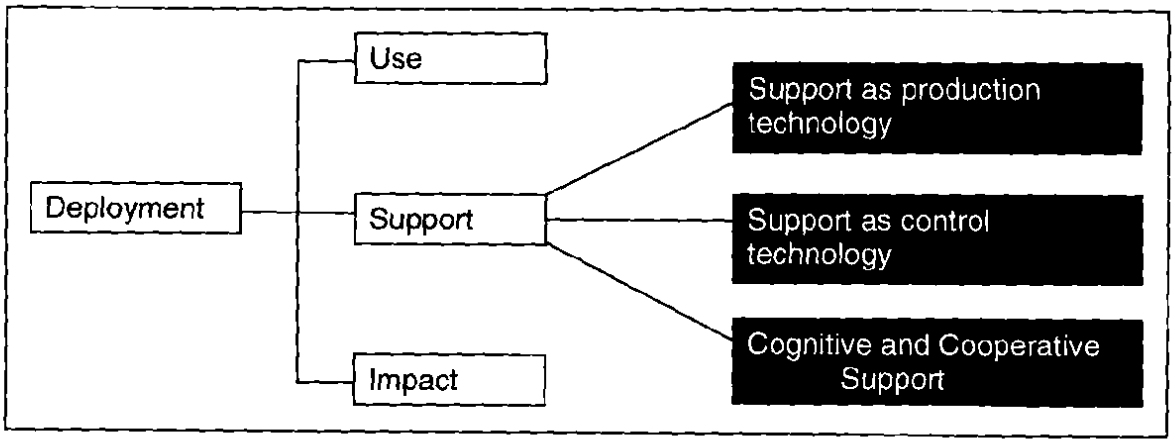

Fig 3.2 Huisman and Iivari (2006: 29-49)

It is stated above that the term *support* includes production support, support in control and also cognitive and cooperative support. Support as production technology provides the project team with tools, techniques and methods to help them with the development. Support as control technology reflects project management support, ways of tracking

progress and keeping to a schedule and budget. Cognitive and cooperative support illustrates the way the methodology guides the way team members work together, and the influence it has on communication and the exchange of information.

3.8 Previous research on the deployment of systems development methodologies at project level

System development methodologies are seen as a core issue in the information software and software engineering field. As discussed above, there are various adopting units when it comes to the deployment of a systems development methodology, namely the organisation, the project and the individual. Although research has been done on systems development methodologies, there is still a lack of research that focuses on the project as the adopting unit. This is illustrated in the summary of empirical studies of systems development methodologies created by Huisman (2000), where only two of the fourteen studies reported focused on the project as the adopting unit.

After a literature review, it was found that the following system development research also focussed on the project as the adopting unit:

According to a study by Fitzgerald *et al.* (2003), advantages can be gained from using a standardized information system development method and tailoring that method to fit the specified requirements of each development project.

Research by Meso *et al.* (2006) supports Fitzgerald *et al.* (2003) as it indicates that the effective matching of a methodology to an application domain influences the effectiveness of knowledge management, and the performing of the systems development work. Matching the methodology also indirectly influences the quality of the information system solution developed. System development is a problem-solving activity that can be classified into two groups, namely weak and strong problem solving approaches. Weak problem solving approaches are general – they can accommodate a large variety of problems but do not assure that the solution will be optimal. Strong problem solving approaches are only useful for certain specialised problems. They have a very specific fit to the application, and the solutions they produce are more likely to be near optimum.

In a study by Howard *et al.* (1999), the use and effectiveness of a strongly typed (strong / specific) methodology was tested and it was found that it might not be as effective as expected. This is counter to Agarwal's (1996) finding who found evidence that strongly typed methodologies were of significant value in improving system designs.

Research done by Rai and Al-Hindi (2000) suggests that systems development methodologies should be narrowed down to a development process model that is specific to the project.

These findings from previous research are not conclusive and illustrate a need for further studies in this field. Further studies in the use of methodologies in South Africa are truly necessary, especially when considering that when this study was done, no literature was found on the use of systems development methodologies by South African software houses at project level.

3.9 Summary

In this chapter a systems development methodology was defined and previous research into the deployment of systems development methodologies at project level was reviewed. These studies indicate that times have changed and that system development is not what it used to be fifty years ago. There is a need for system development methodologies to adapt to an ever changing information system environment and choosing a methodology that fits the application is increasingly important. There are still many arguments against and in favour of systems development methodologies. Some of the arguments mentioned above still cause difficulties, and are therefore still relevant in today's system development environment. Systems development methodologies can have rational and political roles in organisations. Although research has been done on systems development methodologies, there is still a shortage of literature in which the project is specified as the adopting unit.

Chapter 4
Research Method and Research Design

4.1 Introduction

Research on the deployment of systems development methodologies can be based on a large number of research approaches. These research approaches depend on the research question, the research target, the theoretical orientation of the study and the research method used. In this study a qualitative approach was followed. This chapter will begin with a discussion of the nature of qualitative research. This will be followed by a discussion of the research method used in this study.

4.2 The nature of qualitative research

Polkinghorne (2005) explains that qualitative research is inquiry aimed at describing and clarifying human experience as it appears in people's lives. Qualitative data are gathered primarily in the form of spoken or written language rather than in the form of numbers. Possible data sources are interviews with participants, observations, documents, and artefacts. The data are usually transformed into written text for analytic use. Selection of interview participants requires purposive and iterative strategies and these strategies may differ according to the type of research that is being done and the researcher's experience.

Qualitative research uses a naturalistic approach that seeks to understand phenomena in context-specific settings (Patton, 2002). Qualitative research, broadly defined in laymen's terms, means any kind of research that produces findings not arrived at by means of statistical procedures or other means of quantification (Strauss & Corbin, 1990). Patton (2002) also refers to it as the kind of research that produces findings derived from real-world settings where the phenomenon of interest unfolds naturally.

The principal advantage, as mentioned by Seaman (1999) and Bryman (2004), is that using qualitative methods forces the researcher to delve into the complexity of the problem rather than to abstract it away. Therefore the results are richer and more informative.

According to Yin (2003), there are three conditions to consider when choosing a qualitative research strategy:

- The type of research question posed
- The extent of control an investigator has over actual behavioural events
- The degree of focus on contemporary as opposed to historical events

These three conditions were considered when the author decided to use case studies as the qualitative research method for this study. More detail on the alignment between this study and these conditions is given in section 4.3.2

4.3 Qualitative research methods

There are various qualitative research methods. These will be discussed in brief followed by a detailed discussion on the author's choice of research method.

4.3.1 Qualitative research method types

- Phenomenology

According to Hancock (1998), phenomenology literally means the study of phenomena. It is a way of describing something that exists as part of the world in which we live. Phenomena may be events, situations, experiences or concepts. Sanders (1982) explains phenomenology as making explicit the implicit structure and meaning of human experiences. It is the search for the essences that cannot be revealed by ordinary observation. Phenomenology is the science of consciousness or experience. It concentrates neither on the subject of experience nor on the object of experience but on the point of contact at which being and consciousness meet (Sanders, 1982). The heart of phenomenology is to get straight to the pure and unencumbered vision of what an experience essentially is.

- Ethnography

Ethnographic studies entail extensive fieldwork by the researcher. Data collection techniques include both formal and informal interviewing, often interviewing individuals on several occasions, and participant observation (Macauly *et al.*, 2000). Because of this, ethnography is extremely time consuming as it involves the researcher spending long periods of time in the field. This also means that the researcher attempts to interpret data from the perspective of the population under study. The results are expressed as though they were being expressed by the subjects themselves, often using local language and terminology to describe phenomena (Ball & Ormerod, 2000).

- Grounded Theory

Grounded theory was first described by Glaser and Strauss (1967) as a method for the study of complex social behaviour from a sociological point of view. Since then grounded theory has been used as a research strategy in many studies, within different disciplinary contexts. In information systems, examples of studies using grounded theory can be found in Urquhart (1997) where the author applies its techniques to investigate interactional tactics used by analysts and clients in requirements gatherings, and in Lehmann and Gallupe (2005) where the study applies a grounded theory method to establish a first theoretical framework dealing with the structure of international information systems and the dynamics of their development and implementation. The basis of grounded theory is that theory is developed inductively from data. The dynamic relationship between data analysis and data collection is a significant characteristic of the grounded theory approach (Strauss & Corbin,1990).

- Case Study

Like surveys, case study research is one of those research approaches which can take a qualitative or quantitative stance (Yin, 2003).

Hancock (1998) mentions that the case study claims to offer a richness and depth of information not usually offered by other methods. By attempting to capture as many variables as possible, case studies can identify how a complex set of circumstances comes together to produce a particular manifestation. It is a highly versatile research method and employs any and all methods of data collection from testing to interviewing (Seale *et al.*, 2004).

4.3.2 Qualitative research method used in this study

The author decided to use interpretive case studies as the research method for this study. At the beginning of this chapter the author referred to Yin (2003) for the conditions one should take into account when choosing a research method. A more detailed look at these three conditions follows:

- The type of research question posed.

In this study the following research questions were asked: "How" are the software houses developing their information systems and "why" are they doing it that way? Yin(2003) states that if the researcher needed to know "how" or "why", the study would lean to either a case study or a field experiment.

- The extent of control an investigator has over actual behavioural events.

This study does not require control over behavioural events. It is merely a study to describe the way system development is done on project level, therefore the interviews were designed with open, discussion type questions, so that the researcher does not lead the interviewee to a specific answer. A case study is preferred when the relevant behaviours cannot be manipulated (Yin, 2003).

- The degree of focus on contemporary as opposed to historical events.

The focus is on contemporary events, not on historic events. The way information systems are developed today, is what the researcher is interested in. According to Yin (2003), the case study is preferred when contemporary events are examined.

The choice of using case study as a research strategy is therefore supported by these conditions.

Arguments against case studies

One of the criticisms aimed at case study research is that the case under study is not necessarily representative of similar cases and therefore the results of the research are not generalisable (Ruegg, 2006). According to Hancock (1998), this is a misunderstanding of the purpose of case study research which is to describe that particular case or a limited

number of cases in detail. Generalisability is not normally an issue for the researcher who is involved in studying a specific situation. It is more of an issue for the readers who want to know whether the findings can be applied elsewhere. It is the readers who must decide whether or not the case being described is sufficiently representative or similar to their own local situation.

More criticism of the case study method comes from those who say that a researcher's bias limits the validity of a study, and that working with numerical data is the only way to remain removed from the subject being studied. Some ideological frameworks view this inability to remain objective as a limitation of the case study. When doing case study research there is always the possibility that two separate researchers working independently of each other may reach two separate conclusions based on the same data. According to critics, this influences the reliability of the study (Yin, 1993).

Arguments for case studies

Researchers continue to use the case study research method with success in carefully planned and crafted studies of real-life situations, issues, and problems. Multiple case designs have the advantage that its evidence is often considered more compelling and that the study as a whole is regarded as being more robust (Herriot & Firestone, 1983).

The advantages of case studies also include (Soy, 1996):

- Its applicability to real-life
- Its contemporary, human situations and
- Its public accessibility through written reports

Researchers from many disciplines use the case study method to build upon theory, to produce new theory, to dispute or challenge theory, to explain a situation, to provide a basis to apply solutions to situations, to explore, or to describe an object or phenomenon (Iivari, 2006; Soy, 1996; Walsham, 1993). In this study the author will use the case study method to explain, explore and describe a situation, the situation being the way software houses develop information systems at project level with the use of system development methodologies.

According to Langley (1999) and Klein and Myers (1999), interpretive studies rely on data-driven reasoning and not theory-driven reasoning. The focus is to understand the situation, and search for coherence and order. Walsham (1995) gives four different generalizations usually obtained from interpretive case studies:

- The development of concepts
- The generation of theory
- The drawing of specific implications
- The contribution of rich insight

Taking the Walsham (1995) generalizations into account, this study uses case studies as a method to obtain rich insight into the way information systems are developed at project level in software houses in South Africa.

After considering both arguments for and against the use of case studies, the author still decided to use case studies as the research method for this study. The advantages of using case studies are more than the disadvantages, and the author tried to reduce those problem areas described earlier in this chapter, by keeping them in mind when the research design was done.

In Yin (2003), both single and multiple case studies are seen within the same methodological framework. Therefore, in the previous section the author did not separate the two, but would like to state that a multiple case design was used throughout this study (fig 4.1). The multiple case study design as shown in fig 4.1 states that more than one case of the same unit is studied. In this study the unit is the project level and the context is the software house and the outsourcing environment. The six system development projects that were studied during this research are represented as six cases. These projects are discussed in more detail in chapter 5.

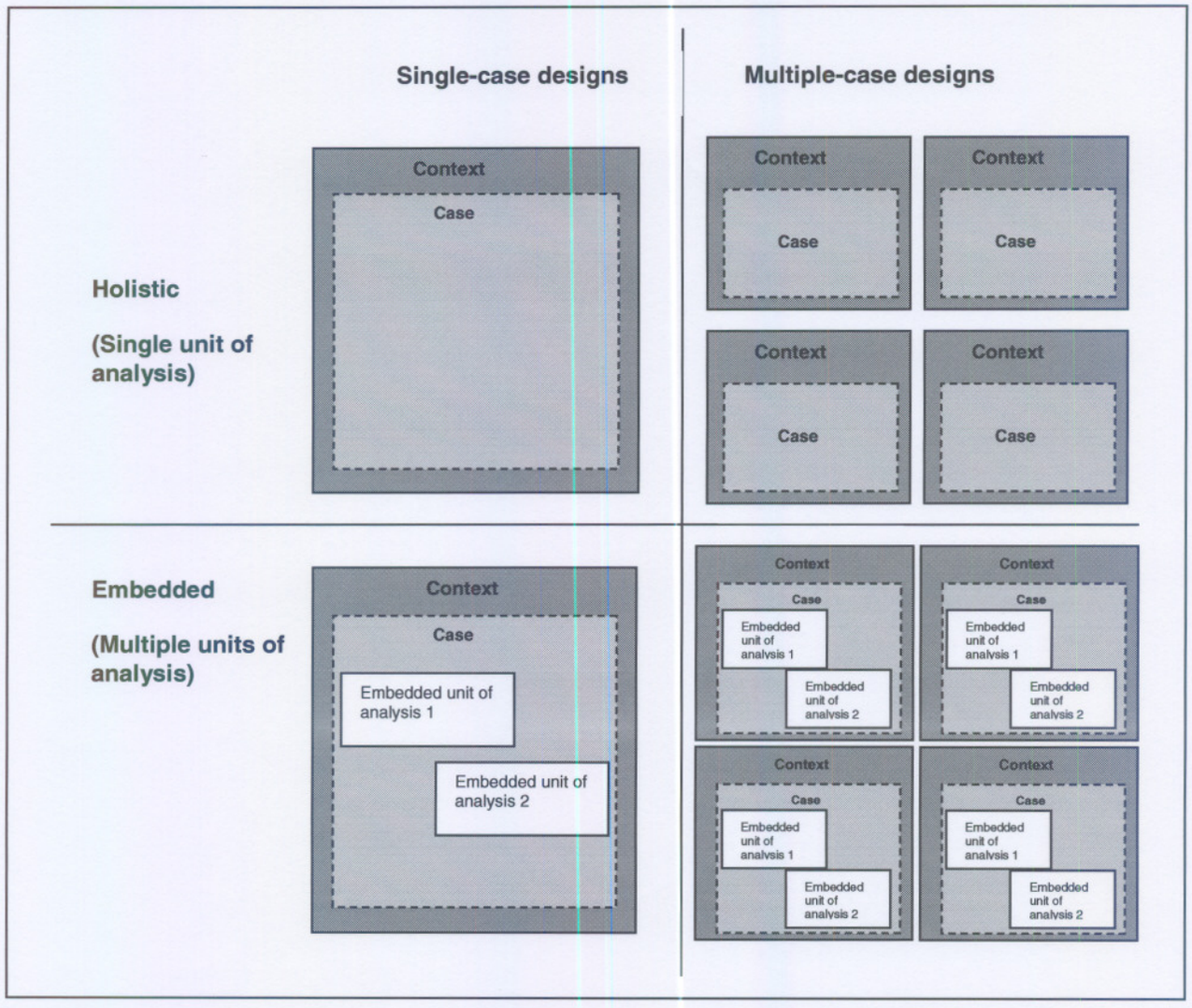

Fig 4.1 Basic Types of Designs for Case Studies (Yin 2003:40)

4.4 Data collection Method

There are various data collection methods. These will be discussed in brief followed by a detailed discussion of the author's choice in data collection method and the implementation of the data collection method.

4.4.1 Qualitative data collection methods available

- Individual Interviews

Interviews are conducted either with individuals or with very small groups. Interviews also encourage a high degree of participation from the interviewees, which generates tremendous detail data (Kimball, 1998). Interviews can be highly structured, semi structured or unstructured (Hancock, 1998).

- Focus Groups

Sometimes it is preferable to collect information from groups of people rather than from a series of individuals. Focus groups can be useful to obtain certain types of information or when circumstances would make it difficult to collect information using other methods of data collection, especially if time is not available to conduct one-on-one interviews. They have been widely used in the private sector over the past few decades, particularly in market research (Hancock, 1998).

- Observation

Observation as a data collection method is very time consuming as the researcher has to be on the study site observing what the subjects of the study are doing. Recording of observation data can be achieved using video or audio taping, in addition to field notes. Observation data also commonly involve recording a 'map' of the setting in narrative or graphic form, for example the layout, staffing and workload during the observation period. It is important to capture the environment and all the surroundings. This is a particularly important collection method for ethnographic studies (Endacott, 2005).

4.4.2 Data collection method used in this study

The author used semi-structured individual interviews to collect his data. The following is a detailed discussion of this data collection method.

- Individual Interviews

Interviews are conducted either with individuals or with very small groups. Interviews also encourage a high degree of participation from the interviewees, which generates tremendously detailed data (Kimball, 1998). Interviews can be highly structured, semi structured or unstructured.

Structured interviews consist of the interviewer asking each respondent the same questions in the same way. Seaman (1999) warns that a tightly structured schedule of questions is used very much like a questionnaire, and that the researcher should be careful, considering that the information could be more efficiently collected using questionnaires.

Semi structured interviews (sometimes referred to as focused interviews) involve a series of open ended questions based on the topic areas the researcher wants to cover (Hancock, 1998). The open ended nature of the question defines the topic under investigation but provides opportunities for both interviewer and interviewee to discuss some topics in more detail. Seaman (1999) states that if the interviewee has difficulty answering a question or provides only a brief response, the interviewer can use cues or prompts to encourage the interviewee to consider the question further. In a semi structured interview the interviewer also has the freedom to probe the interviewee to elaborate on the original response or to follow a line of inquiry introduced by the interviewee. In semi-structured interviews, only some of the questions are predetermined, and interviews are always conducted face-to-face. The interviewer works from a series of open-ended, lead-in questions to direct the respondent towards certain issues of interest without leading the respondent towards a pre-determined or desired response (National Forest Foundation, 2006). This is supported by Hancock (1998) who states that semi structured interviews tend to work well when the interviewer has already identified a number of aspects he wants to be sure of addressing. The interviewer can decide in advance what areas to cover but is open and receptive to unexpected information from the interviewee.

Following is a list of advantages and disadvantages of semi-structured interviews as described by the National Forest Foundation (2006):

Advantages of semi-structured interviews

- Flexibility: interviewer may alter sequence in which questions are asked, probe or otherwise seek clarification of responses, and follow-up on unexpected responses;
- Comprehensive: provides breadth of information;
- Intensive: provides depth of information;
- Personable: puts the respondent at ease and may provide access to personal information that would otherwise be unavailable; and
- Comparison of responses of individual respondents is often possible.

Disadvantages of semi-structured interviews

- Data collection often requires travel and can be expensive;
- Data analysis is often complicated and time consuming; and
- Data may contain irrelevant information.

Unstructured interviews (sometimes referred to as "depth" or "in depth" interviews) have very little structure at all (Hancock, 1998). The interviewer goes into the interview with the aim of discussing a limited number of topics, sometimes as few as one or two, and frames the questions on the basis of the interviewee's previous response. Although only one or two topics are discussed they are covered in great detail.

Walsham (1995) states that a key issue for all interviewers is the balance between being too passive and over directive. If the interviewee is too controlling the data obtained will lose much of its richness.

After having considered all the options, advantages and disadvantages, the author chose to use semi-structured face-to-face interviews as the primary data collection method. According to Seaman (1999), these semi-structured interviews include a mixture of open-ended and specific questions, designed to elicit not only the information foreseen, but also unexpected types of information. The use of semi-structured interviews also guides the interviewer to obtain the balance, mentioned (above) by Walsham(1995), between excessive passivity and over-direction.

The software houses involved in this study were randomly selected out of companies in Gauteng, after which a face-to-face interview with a member of each of six projects was arranged. The interviews were conducted in September 2005. There are two reasons for choosing software houses in Gauteng. The first is that Gauteng is the economic driver of South Africa and therefore also an obvious trendsetter in the use and development of information systems. Secondly, Gauteng is within a reasonable travelling distance, which is a solution to the traveling disadvantage mentioned by National Forest Foundation (2006).

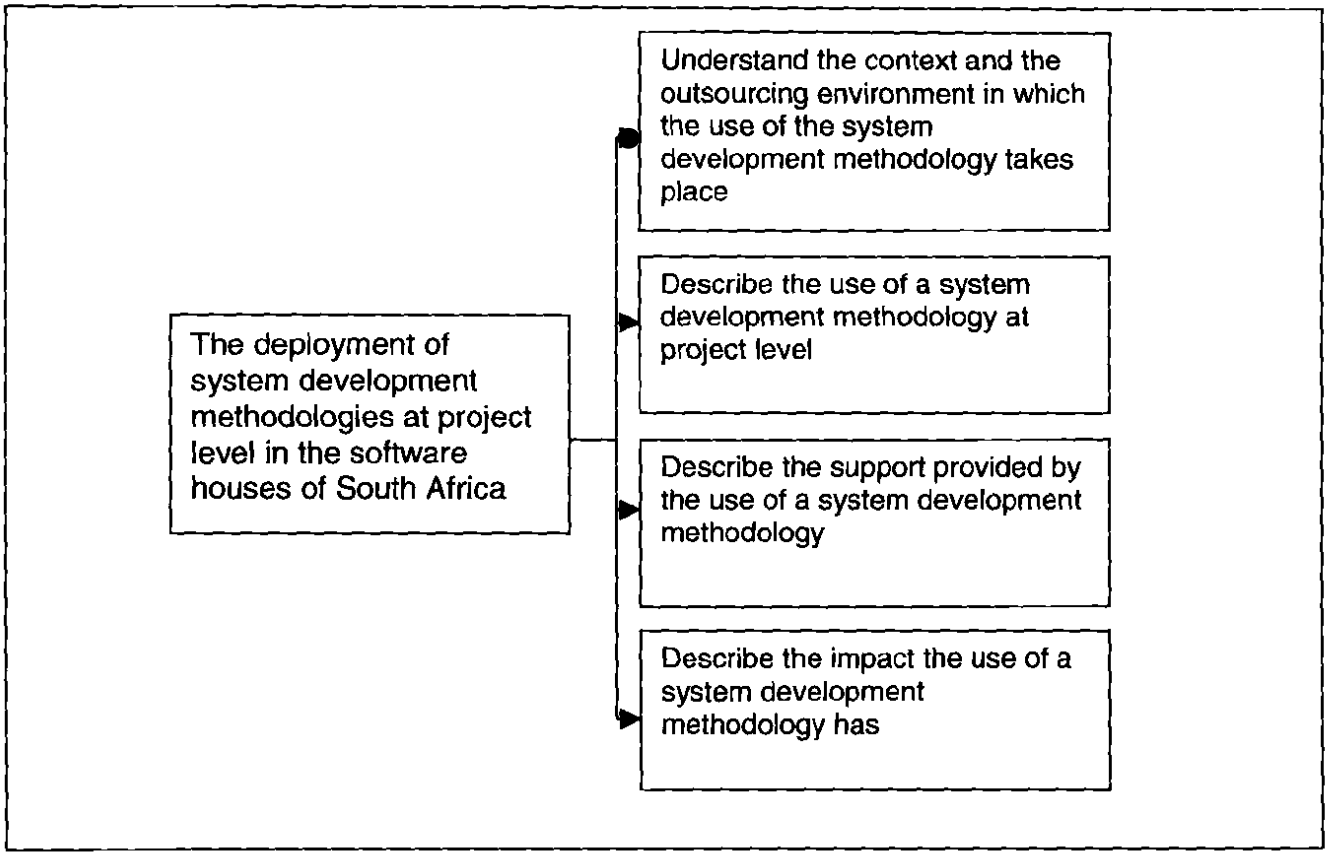

Fig 4.2 Illustration of research design

The researcher created his set of questions based on the scope of the study as illustrated in fig 4.2. The author made sure that all the different fields in the study were covered equally by the questions asked. These questions were tested with a test interview and changed to make sure that they were not leading or intimidating in any way. The data obtained from the test interview were not included in the findings of this study.

These semi-structured interviews were all guided by interview protocol and the interviews dealt with the following issues:

- Company background – To be able to see the big picture, it is very important to take the context of each scenario into account.
- Outsourcing environment – Explaining how the environment in which they work has changed and how that might influence the way they use system development methodologies at project level.

- Project description – This study particularly targeted the project level of system development methodology use.

- How the system was developed: The author did not specifically refer to systems development methodologies as that would have led the interviewee into that field. Instead the author questioned the interviewees on how the system was developed and the process that they followed. Then, from their answers, the author was able to derive information regarding the use of systems development methodologies and the impact it has.

A total of six projects (cases) were targeted. The average length of each interview was approximately one hour and all the interviews were recorded using a laptop and voice recording software called MP3 Recorder Pro. The interviews were all conducted on the premises of each software house. To commence analysis the author transcribed the interviews. These transcripts were then used as the primary source of information for the analysis phase.

4.5 Data Analysis Methods

4.5.1 Qualitative data analysis methods available

Hancock (1998) mentions that analysis of data in a research project involves summarizing the mass of data collected and presenting the results in a way that communicates the most

important features. Quantitative research analysis involves things like the frequencies of variables, differences between variables, statistical tests designed to estimate the significance of the results and the probability that they did not occur by chance (Hancock, 1998). All this is done basically by counting how often something appears in the data and comparing one measurement with others. At the end of the analysis, the author has a mass of results and the major findings.

In qualitative research the author is interested in discovering the big picture and can use different techniques to find it. As in quantitative research, there may be some data which are measurable but for the most part the focus is on using the data to describe a phenomenon, to articulate what it means and to understand it (Seaman, 1999).

According to Hancock (1998), the basic process of analyzing quantitative and qualitative data is the same. The researcher starts by labelling or coding every item of information so that he can recognize differences and similarities between all the different items. Coding qualitative data requires different techniques. The qualitative researcher has no system for pre-coding and so needs a method of identifying and labelling (coding) items of data which appear in the text of a transcript so that all the items of data in one interview can be compared with data collected from other interviewees (Hancock, 1998). There are two types of data analysis methods viz. content analysis and tape analysis These are described bellow. The procedure is the same whether the qualitative data have been collected through interviews, focus groups, observation or documentary analysis since it is concerned with analyzing text.

- Content analysis

Content analysis, according to Hancock (1998), is a procedure for the categorization of verbal or behavioural data, for purposes of classification, summarization and tabulation. The content can be analyzed on two levels. The basic level of analysis is a descriptive account of the data: this is what was actually said with nothing read into it and nothing assumed about it. The higher level of analysis is interpretative: it is concerned with what was meant by the response, what was inferred or implied.

Content analysis involves coding and classifying data. The basic idea is to identify from the transcripts the extracts of data that are informative in some way and to sort out the important messages hidden in the mass of each interview (Barry, 1998).

Computerized data analysis supports the content analysis. Software has been developed which can help with the analysis of qualitative data. An increasing range of packages are available, each one with different features and some are more popular with researchers than others. The packages, if properly used, can save the researcher a great deal of time but a fair amount of human input is still required to identify and check categorization.
Some of the best known software packages are listed below:

- ATLAS/ti
- NVivo
- NUD*IST

Two methods exist for doing content analysis, constant comparison analysis and cross-case analysis.

Constant comparison is a content analysis method which specifies that the investigator will continually examine and compare elements throughout the whole research project. These include elements such as data instances, emerging categories and theoretical propositions (Carvalho *et al.*, 2005). This method of analysis is often used in grounded theory and therefore was not suited for use in this study.

The other technique, cross-case analysis, is a search for patterns that keeps investigators from reaching premature conclusions by requiring that investigators look at the data in many different ways (Palmquist, 1997). Cross-case analysis divides the data by type across all cases investigated. The researcher then examines the data of that type thoroughly. When a pattern from one case is corroborated by the evidence from another, a stronger pattern is formed. When evidence conflicts, deeper probing of the differences is necessary to identify the cause or source of conflict. In all cases, the researcher goes through the data thoroughly to produce analytic conclusions answering the original "how" and "why" research questions (Palmquist, 1997).

- Tape analysis

Accoring to Hancock (1998), it is advisable, if at all possible, to analyze qualitative data using transcribed records of data. If transcripts of recorded interviews are not available it is possible to carry out tape analysis. This involves replaying the tape recording of an interview and making notes of relevant and interesting data rather than full transcripts. It is much less time consuming than transcript analysis but it has a number of disadvantages. The procedure is open to researcher bias as the researcher is likely to make notes of information that is immediately recognizable as useful and potentially relevant information may be overlooked. The quality of the analysis may also lack depth and comprehensiveness. This can compromise the accuracy of data and therefore the reliability and validity.

4.5.2 Data analysis method used in this study

Cross-case content analysis

The author decided to use cross-case content analysis as the data analysis method for this study. Cross-case analysis was used in a number of other studies including Iivari (2006) and Seaman (1999). Eisenhardt (1989) suggests several useful strategies for cross-case analysis, all based on the goal of looking at the data in many different ways. In this study the author studied the data in the cases as mentioned below, and to obtain a deeper analysis the cases were also grouped in the sizes of the projects, and the type of methodology used.

The author used ATLAS/ti to assist in the process of categorizing the information obtained from the transcribed interviews. The decision to use ATLAS/ti in this study was mainly based on its availability. The analysis process, using ATLAS/ti software is very similar to doing the analysis by hand. It is just a lot more organized and easier to categorize related ideas and follow a specific line of thought. The following are the general steps that the author took.

- Create a project, an "idea container," meant to enclose your data, all your findings, codes, memos, and structures under a single name. This is called a "Hermeneutic Unit" (HU). This was saved as “M Interviews”.

- The next step was to assign documents (transcripts of the interviews) as Primary Documents (PD) to the HU. These data files became the source material for the project.
- The author then read and selected text passages that are of further interest, assigned key words (codes), and wrote memos that contain the thinking about the data. This is called the Textual-Level working phase.
- Then the comparison of the data segments starts. Based on the codes assigned, each primary document is compared to the other primary documents.
- Organize PDs, codes, and memos using "Families".
- Build semantic, prepositional or terminological networks from the codes created. These networks, together with codes and memos, form the framework for the emerging findings.
- Finally, compile a written report based on the memos written throughout the various phases of the project and the networks created.

A cross-case matrix was developed for each of the code families generated in ATLAS/ti. This matrix is presented in Table 4.2. This matrix allows analysis down rows and across columns to see what similarities and differences there are. In this way, it was possible to compare data from the six cases and to discern patterns or themes. To thoroughly analyze the data the author first studied each case intensively, transcribing the interview, coding and classifying the data. Then two of the cases in the matrix were studied together to see what similarities there were, and what differences exist between them. After going through the first two cases, the author drew up some propositions that one might draw if these were the only two units in the data. A list of supporting evidence was produced. The next step was to compare these similarities, differences and propositions to a third case in the matrix. Based on those data the author either accepted or modified the previous propositions and added new themes or patterns. This process was repeated to identify as many similarities and differences as possible, and until all six the cases were analyzed. The results as discussed in chapter 5 are the propositions that were supported by the data.

Code	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Specialization	Software design and development	Wide range of solutions	-	SAP solutions, but across the board Mainly work in the chemical space	-	Yes Business solutions for human capital management SAP – HR
Size of company		13 employees 9 work in IT small to keep sustainability	More than 30 employees 4 Abbab programmers Rest work on customization	Changes often 30 employees and extra contractors All in IS except 1 PA	-	30 Employees. Most work on IS
Task profile	50% new development and 50% maintenance	Mainly work on new systems development	40% new development and 60% customisation / maintenance	Was 70% maintenance and 30% new Is shifting to a 50%-50% divide More growth in new development	40% New development and 60% maintenance	50% new development and 50% maintenance
Project	Credit card verification system Changing of the credit card system so that it works with a pin code	Royalty management project Managing film royalties from different studios around the world Throughout the film's lifecycle	Implementation of the production leg of SAP Not just normal SAP – integration with 3rd party software	Reimplementation of all IT systems at a chemical company	Rush pharmaceutical New development Changing from VIP payroll to SAP payroll Migration of data to the new system. Integration with the third party	Trans Namib project A full ERP implementation. Replacing a paper based system Needed consolidation and integration
Project size	Very Small	Small	Not big / small	Very Big	Small	Average
Project budget	Can't remember the budget	1.5 million	250 000	12 million	Not a clue	3.4 Million
Project status	Complete	In progress	Complete	Still in progress	Complete	Complete
Project source	Started by fixing the old system, forced to new development	Relationship with another company	Was already a client	Tender against other products	They got the project through marketing	Partnership with Siemens New developments come from the business solutions side of the company

Project team size	1 man project	Average 3 people Maximum 6 people	6 people	Max 20 people	5 people	6 people Experienced people come and go while the juniors stay from start to finish
Characteristics of project team		Team lead is very experienced developer Combination of experienced and inexperienced people	-	-	-	-
Project team morale	N/A – 1 man project	Difficult to maintain morale That's why the team worked on an rotating schedule	They worked together very well	-	The group worked together quite closely Everyone knows the bigger picture Integration is very important	-
Challenges	Client expectations Users did not want the changes Communication was difficult – client was based in the UK	Changing user requirements Integration Changing environment	No challenges Most successful project	Many challenges Change management was a Big problem Scope Creep User availability	None It was his best project But he mentions client expectations	Change management was the main problem Clients culture was not ready for an ERP system Computer illiteracy was a problem Communication gap Generation gap
Project management	1 man project Used software tools to keep track of the different versions Internet based log	Team lead – project manager Very open management style Tool: Used Excel for time management	Project Plan based on ASAP ASAP resulted from best practice	High level timelines. Project manager – control Tools: MS project	Very important Used remote access with diginet lines to minimize travelling Tools: MS project was used	Very important Project manager. Personalities might clash, so the project manager has to choose the team and manage any situations that might occur Tool: MS Project was used Meetings once a week

Problem management		Adaptable methodology Competent people	-	Top down approach which he led	Client expectations are managed by the process Regular signoffs and reports are essential	People are the main solution Senior consultants act as change agents. The users became excited and supportive after some work
Relationship management	Regular conference calls Daily emails and issue reporting	Regular meetings	Project manager in charge – he is normally an experienced developer After completion – still involved for about 1 month All problems normally appear at month end User champion (trained, client) then takes charge of problems	Business people are in charge Meetings – once a week After completion – still involved on a high level New problems – are treated as new phases	Functional consultants are the client facing people	Project manager builds relationship with client Account manger (on site) becomes part of the client Gains trust and does tests Meetings once a week
Advantages of Relationship management	-	No surprises Keep your finger on the pulse	The user champion is on site He was trained- knows the system and can explain problems in detail	It keeps everyone up to date (informed)	-	Trust is built Account manager knows what to test and how to explain the problems. On site, so he can fix errors on the spot
Disadvantages of this way of Relationship management	-	Time consuming	User champion can keep his knowledge to himself and not share it	-	-	Client has your employee Management politics Not easy to manage
Outsourcing		Pro outsourcing Client and SH must fit	Pro outsourcing Cheaper for small companies	Pro outsourcing Especially for SME's	Inhouse if you've got the knowledge base Recommends outsourcing the entire IT infrastructure	Pro outsourcing Especially for SME's

Advantages of outsourcing	Higher productivity	Higher productivity Don't worry about personnel turnover More options on who works on the project	Don't worry about personnel turnover Developers are normally of a higher quality SH has a lot of experience They have seen different scenarios – more solutions	Don't worry about personnel turnover Many service providers Big knowledge base More solutions to choose from.	Bigger knowledge base 1 point of contact SH has to coordinate everything	Bigger knowledge base SH is much stricter on doing it right IS is top priority for the SH Client can focus on other strategic issues Costs are spread across many clients Can supply a total HR solution, not just an IS system
Disadvantages of outsourcing	-	Risk that client's priorities and SH's priorities may differ Intellectual property gets lost Risk of choosing a bad SH	Loyalty Intellectual property is lost The in-house people don't know the system	Risk of choosing a bad SH	-	Companies need to be careful on what they outsource
How to choose the right SH	-	Look at the SH's track record Make sure the client and the SH fit Use SLA's	-	Relationship at management level is vital The SH must have interest in your type of business	-	-
Process	Loosely based on the Iconics process Mentioned UML, and flow diagrams Breaks the process into phases	Loosely based on XP Somewhere between XP and Agile Breaks process into phases Reiterated and reassessed	Emphasis on project plan and ASAP. Breaks process into phases. Knows the phases. Linear process based on traditional SDLC. ASAP resulted from best practice.	ASAP methodology Breaks process into phases Tools: Solution Composer to help define the scope ASAP takes scope as input, then it can generate the different tasks and estimate durations for the different phases Does not know all the phases	Broken into phases Loose process Lots of documentation Abbab workbench	Broken into phases ASAP methodology SAP templates Training

Experience of use	Short – learned it at university	Long – Engineering background	Long: 7 Years	Long	Average 3 Years	Long Believes very strongly in experienced people, juniors are assisted by seniors
Requirements analysis	Better analysis leads to better quality code	Better analysis leads to better functionality Must be able to handle changing user requirements Start on common ground and work from there Use system and business analysts doing interviews to get the necessary information	User must understand what system will do Use blueprints to show changes and outcomes Can absorb changes	Always difficult Tool: Solution composer helps to determine the scope Change way clients think New changes are handled as 2nd or 3rd phases	Speak to clients Kick off meeting	Workshops with users Have to REALLY listen, need experience RACI matrix
Reasons for use	It helps implementation logic (direction) Improves accountability Cut the parts that only add complexity	It fits the people working on the project It fits the client Not forced	Best to work with a plan. It gives direction. It supports project management. It is proven practice. Not forced.	When doing SAP certification the developers learn ASAP To be a SAP partner the employees must be certified Not forced Fits the software	Blueprints help a lot Keeps client up to date – a lot of sign offs (Communication) Documentation is very important	It fits the SAP environment (software) Have tried other methodologies, but error is much higher Documentation is very important
Documentation	Very important Especially for expandability and upgrades Tool: Used to use "Rational Rose and Visio"	Documentation is very important Keeps everyone informed Very important when handing over the project	Knows it's important but does not like to do it It takes a lot of time ASAP has tools that encourage documentation It's not forced	ASAP improves documentation	Very important Not forced to do it Prefers to do it so that he knows what and why he did something Blueprints	Very important
Advantages of use	Good design results in better code Gives structure	Prototypes keeps customers happy Adaptability, changing user	Project plan is vital. Gives direction	Fits the software Big knowledge base – all SAP people use it Gives direction and	Prevents scope creep Improves communication	Not one delay Better communication

	to the process Accountability Gives direction	requirements Accountability Not strict, so it does not damp initiative Guided by the framework (direction)	Supports project management.	focus Improves communication in the team (everyone thinks the same)		
Impact use has on system	Positive impact More design leads to better code Systems are more expandable and upgradeable The methodology does not influence user satisfaction – they just look at the interface	None. Better analysis leads to better functionality Quality is result of the people, not the methodology	Positive impact on quality Positive impact on system Positive impact on functionality (planning)	Positive impact on quality Positive impact on documentation	-	Easier to maintain and enhance
Impact use has on process	Takes longer because of proper designing More expensive in the short term Cheaper over long term	Positive impact Faster Better documentation Cheaper	Positive impact Better productivity Better chance that it will be on time	Positive impact Gives team focus Improves communication Saves time Better documentation	Prevents scope creep Better documentation	Gives direction (focus) Better documentation Not one delay Improves productivity
Maturity and Certification	-	Maturity is not important ISO not important Interested in Microsoft certification	They do not have ISO certification He does think that ISO might be important when choosing a SH but not very	Structure of company changed too often to obtain maturity ISO standards are not important SAP certification is very important	-	ISO standards not important Certified SAP partner
New technologies	-	-	Gets information from SAP	Introduced through ramp-ups and workshops by SAP	-	-

Table 4.2 Cross case matrix of the six cases.

4.6 Confirmation of findings

4.6.1 Qualitative finding confirmation techniques available

There are various methods to confirm or strengthen the proposition after it has been generated from the data of a qualitative study. The goal is not to prove a particular proposition, but to build a "weight of evidence" in support of it (Seaman, 1999; Golafshani, 2003).

- Validation of techniques

Seaman (1999) states that an important way to help confirm a quantitatively generated proposition is to ensure the validity of the methods used to generate it. The basic idea is not only to ensure that the data collected are accurate, but also that they are not recorded in a way that is only understandable to the researcher.

- Triangulation

The basic idea is to gather different types of evidence, possibly from different sources; they might even be collected using different methods, have different forms, or come from a different study altogether.

- Negative case analysis

Negative analysis as stated by Judd (1991) is another analysis tool for helping confirm hypothesis and propositions. The process involves an intensive search for any evidence that might contradict the generated proposition.

- Replication

Replication is a powerful but expensive tool for confirming findings. It is done by doing a study on another subject that fits the conditions of the findings. The data do not have to be collected or analyzed in the same way (Seaman, 1999).

- Member checking

Seaman (1999) states that member checking is particularly well suited to most studies of software engineering. Member checking is the process of getting feedback on the findings from the subjects who provided the data in the first place (Lincoln & Guba, 1985). An added advantage of presenting the findings to the subjects is that they feel more involved in the process.

4.6.2 Confirmation of findings technique used in this study

Validation of technique

In an effort to validate the technique used to obtain the findings. The author first conducted a test interview and adapted his style to make sure that he was not leading interviewee or intimidating in any way.

Member checking

In this study member checking was used to confirm the findings. Erlandson *et al.* (1993) and Lincoln and Guba (1985) both state that member checking is a particularly important method of increasing the credibility of qualitative research. During the data collection stage it was mentioned to the interviewees that their participation would be needed in a member checking exercise. The interviewees agreed to this, and showed a big interest in receiving the findings of the study. According to Annon (2002), member checking following data collection consists of reporting back preliminary findings to respondents or participants, asking for critical commentary on the findings, and potentially incorporating these critiques into the findings.

The completed written data analysis and results sections of this study were sent to all the participants to ask for their feedback. All but one of the participants provided feedback through the member-checking process. The participants who did respond confirmed that the analysis and the results did represent their overall experience.

4.7 Summary

In this study the author used a qualitative research method to study the deployment of system development methodologies at project level in the software houses of South Africa. Multiple case studies were done, whereby the data were collected using semi-structured interviews. These interviews were transcribed and cross-case content analysis was done with the help of ATLAS/ti. The author then followed a member checking process to confirm the findings of the study.

Chapter 5
Findings

5.1 Introduction

In this chapter the author will present the findings of the study. The author will start by re-stating the aims and objectives of the study, after which he will describe each case used in this study. Thirdly the environment related findings will be discussed. These include, challenges faced by software houses, how project management and client relationship management take place in software houses and outsourcing perceptions of the project team members. Fourthly the findings with regard to system development methodologies will be discussed. This section will include a description of the use, support and impact of the methodologies on project level.

5.2 Research aims and objectives

This study extends previous research of Huisman and Iivari (2002a, 2002b), who studied the deployment of system development methodologies at organisational and individual level. This study focussed on the use and effectiveness of system development methodologies at project level in the software houses of South Africa. It also examined the environment in which these software houses operate. The outsourcing of system development and even entire information technology infrastructures is a trigger that could stimulate system development and the use of methodologies at software houses. Thus, through this research the researcher would also like to determine how the project team members in the software houses perceive the outsourcing of system development, and the challenges and advantages that the environment entails.

As mentioned in chapters one and three, there are various arguments for the implementation of systems development methodologies and also against the implementation of systems development methodologies (Avison & Fitzgerald, 2002). These arguments may influence the deployment of systems development methodologies in South African software houses. Therefore this research, based on the work by Huisman and Iivari

(2000a, 2000b), studies the following three aspects of the deployment of a system development methodologies at project level in the software houses of South Africa:

- The use of systems development methodologies at project level.
- The perceived support that the systems development methodologies provide at project level. These include production support, support in control and also cognitive and cooperative support.
- The perceived impact that the use of systems development methodologies at project level have on the system and the development process.

5.3 Project descriptions

In this study the author randomly chose six projects at software houses in South Africa as the subjects of his study. These projects are described below:

Case 1

Project: Credit Card verification system

The aim of this project was the development of a credit card verification system for a UK based company. The UK government put legislation in place which stated that all the credit cards had to move from the magnetic strip to the integrated chip and pin, so when you do a credit card transaction you have to enter a pin as opposed to just signing a name. This shifts the liability from the shop owners to the actual banks. It was a very small project which was completed in about six months and there was only one South-Africa based person working on the project.

This project started as maintenance on an old system, but was forced to a total new development because of technology constraints on the old system. The developer who worked on this project used his own in-house developed methodology.

Case 2

Project: Royalty management system

This project involved the development of a system for the managing of film royalties from the different studios around the world. It follows each film through its entire lifecycle from when it is first shown in a cinema to where a film goes out on DVD and video. You have different royalties for digital and analog films and then they are ultimately sold to

consumers in retail shops. Through this lifecycle certain royalties need to be paid back to the studios. This project started in June 2004 and at the time of the interview the completion was set on July 2006. The budget for this project was about 1,5 million rand. This was a relatively small project, with a project team of about 3 to 6 members. The development process of this project was loosely based on XP, almost a combination of the XP and Agile methodologies.

Case 3

Project: Development and implementation of an enterprise resource planning system for production

This project was done for a pharmaceutical group. It involved the implementation of the SAP production system and included the development of an integration hub for the third party sales system to integrate easily with the SAP system. The pharmaceutical group used an external agent to do their sales, and the sales information was necessary to make accurate production predictions on the SAP system. This was not a big implementation, but was quite specific and included the integration with the third party system. The project took about 6 months to complete and had a budget of approximately R250 000. The project team, of about six people, used the ASAP methodology.

Case 4

Project: Total enterprise resource planning system development and implementation

This project was done for a chemical company. It involved the full re-implementation of all their information technology systems. All their old systems were to be replaced with a fully integrated SAP system which would cover human resources, finance, production, client relationship management and business warehousing. It was a very big project that started the 1st of July 2005; it was still in progress at the time of the study, with the completion date set on the end of April 2006. The budget for this project was about 12 million rand and the project team consisted of a maximum of 20 people. The ASAP methodology was used in this project.

Case 5

Project: The development and implementation of an enterprise resource planning system for human resources and payroll

This project involved the implementation of the SAP HR system for a pharmaceutical company. This SAP system manages and integrates the payroll of the client company. It was a small project that included the migration of the information from the previous payroll system to the new SAP system. The project team consisted of about 5 people and the project was completed in approximately 9 months. The project was completed successfully by using the ASAP methodology.

Case 6

Project: Total enterprise resource planning system

This project involved the implementation of a total enterprise resource planning (SAP) solution for a Namibia based transport company. The main focus of this project was the HR payroll system. It was an average size project which started at the beginning of April 2004 and concluded in March 2005. The budget for the project was about R3,4 million. The project team, with an average of six members, used the ASAP methodology during the project.

5.4 Environment

In this study the environment in which software houses operate was studied. Certain recurrent themes became apparent in the material gathered from each case. Various challenges faced by the software houses were mentioned by the interviewees. These were then categorized by the author into two groups, namely user related challenges and environment related challenges. After discussing these challenges the author will discuss the rest of the findings obtained from this study in terms of the following categories: project management in software houses, client relationship management in software houses and perceptions of project team members regarding outsourcing.

5.4.1 User related challenges

User related challenges as categorised by the author are those problems that are directly related to the users of the system being developed. These types of challenges occurred in five of the six cases (1, 2, 4, 5 and 6). They are:

- Changing user requirements

Changing user requirements were frequently mentioned as a big problem, for example *"I would say the two main problem areas that we found were changing user requirements and business models"* (Case 2). When user requirements keep on changing, *"...then a client comes back and keeps on coming back again and again.... shifting the goalposts all the time"* (Case 5). This makes it very difficult for the team to know what to build. When the requirements constantly change it becomes a big challenge to finish the project in time, as illustrated by the following remark: *" ...that you've got four days to develop this in and instead it takes you 2 weeks"* (Case 5).

According to Case 6, making sure the client knows what they want is a very difficult part of system development. If it is not done properly and signed off, handling the changing user requirements will become a big challenge. Another aspect of changing user requirements is that of extending user requirements. When users add more requirements to a project they can cause scope creep. Scope creep can cause many challenges and defining the scope of a project is one of the first challenges a project team faces. This phase has a tremendous impact on the rest of the project and is therefore extremely important. Uncertainty tends to arise when the boundaries of the system are constantly shifted. *"The big problems are normal scope creep and user availability"* (Case 4 (Translated)). According to Case 6 it is very important that everyone knows the scope of the project otherwise it becomes more difficult to manage scope creep: *"All the consultants need to familiarize themselves with the project, know what is in scope so they can manage scope creeping and all that".*

The author found a clear difference in the way the smaller projects (cases 2 and 3) handled the challenge of changing user requirements compared to larger projects. Case 4, which was a very big project with a strict management style, did not allow for changing requirements: *"I am quite strict when it comes to that, except if it will totally influence the*

way they do business so that they cant run it. But otherwise I'll handle it as a 2nd or 3rd phase, otherwise you'll never get it done" (Case 4 (Translated)). In contrast to the previous view, Cases 2 and 3, both small, tried to accommodate the user by absorbing most of the changes: *"We normally do put quite a lot of latitude in for changes and we try to absorb, I would say sixty seventy percent of changes in scope, whether they are in scope or not"* (Case 2).

- Client expectations

With prototypes and interface designs clients are fooled into thinking that a fully functional system can be delivered very quickly. This creates the problem of managing the clients' expectations of when the fully operational system should be finished, as illustrated by the following remark: "*I mean the hardest part of software development is managing the clients' expectations"* (Case 1). It also tends to lead to clients expecting more than they asked for. According to Case 5, clients tend to expect that the system can quickly be changed or that a new part can immediately be added to the system, just by phoning the developer and asking that he does it: *"We have a lot of clients, and a lot of the time they will just be phoning me up and say: Hey "Mr. X", can you do this, or whatever".*

- User availability

When the primary users of the system are not available to give their input, it makes it extremely difficult for the development team to develop a system that will meet their requirements. Case 4 (translated) mentioned that: *"The big problems are normal scope creep and user availability".* Also mentioned by Case 4 is that it is normally the people who need very specific functionality from the system that are to busy to participate in the requirement analysis phase. Another part of this problem is that when working with off-shore clients, user availability becomes even more difficult. When asked about problems experienced during the project, Case 1 replied as follows: *" Besides the guys in the UK, no, no problems at all".* Although offshoring has become increasingly popular (Avison & Fitzgerald, 2006), it tends to pose a user availability challenge to software houses.

- User resistance

The system that is being developed will change the way the users use to work. However, a lot of users are happy with the current system that they are using, and when a new system

needs to be implemented they tend to resist it. They do not participate in making the process run smoothly. For example, Case 6 noted the following: *"... the guy was very anti, because he was happy with his payroll system".* This resistance by users makes the implementation of a new system increasingly difficult.

- Computer literacy /User expertise

Another user related challenge faced by software houses, is the lack of computer literacy among some users. Some users are not familiar with the environment in which the new system operates. This is illustrated by the following remark: *"Then, computer literacy was an issue... because a lot of people do not have the expertise... but I think the biggest problem is actually the lack of experience in that environment"* (Case 6).

- Communication and generation gaps

According to De Villiers (2002), communication is one of the factors that often lead to project failure. Supporting De Villiers (2002), this study found that problems tend to arise when users communicate their needs to the development team, especially if there is a big age difference between the development team and the users. This is illustrated by the following remark: *"...there is a communication gap, and if you look at for example the generations you have a guy here who is in his thirties / forties and you have one who is in his twenties. If you ask me something, I will give you an answer. You might mean something very different to what I understand, so it's a communication gap"* (Case 6). According to Case 4 (Translated), communication problems are sometimes caused by the fact that decision makers are often busy. They do not take time to verify the requirements and specifications, and in the end they receive a system which differs from what they expected initially: *"Because the decision makers are the ones that are constantly busy, and don't really listen to your explanations"* (Case 4 (Translated)).

5.4.2 Environment related challenges

Environment related challenges are those types of challenges that are directly related to the environment in which the software house operates. These environment related challenges occurred at four of the six cases (2, 3, 4 and 6).

- Changing environment

As mentioned in chapters one and two, the environment in which project development takes place is constantly changing. Businesses change and adapt in response to these changes, and therefore the business models are also subject to change: *"I would say the two main problem areas that we found were changing user requirements and business models"* (Case 2). Another challenge faced by software houses, is the fact that the industry itself is changing a lot. Takeovers can change the structure of a software house dramatically. This was the case at one of the software houses, where the software house changed very often because of takeovers by other companies. In 1998 the original company unbundled and sold all their business units. Company X was then started and sold to a bigger group. Up to 2005 this company was a business unit in the bigger company. Then the company was split into three parts and today Company X exists as an independent company. These changes in the structure of software houses can result in many challenges when working on a project, including the change of specialisation area as was the case in the company discussed above.

In studies by Hamel and Parahalad (1994) and Fitzgerald (1998), it was found that the environment could be described as increasingly turbulent with change being the norm rather than the exception. Successful organisations are often thought to be those that are capable of dealing with such change and the opportunities it present.

- Integration with a 3rd party system

The projects developed by software houses often have to be integrated with 3rd party systems. This integration process can be a very tricky part of a system and should not be underestimated: *"...our system had to cater for the integration into other applications like general ledger applications and billing applications. I would say those were the challenges we had on this project".* (Case 2). Case 3 (Translated) also described how their project had to cater for the integration with a 3rd party sales system: *"...customisation and of course integration with a third party product.... Because there was sales integration, we had to speak to the sales people as well, to determine how the sales process works".* Communication becomes more complex, because besides the normal communication between the software house and the client organisation, the 3rd party also has to be involved in the process.

This is supported by Weakland (2005), who states that the second biggest concern for software houses as well as client organisations is the possible reduction in effectiveness as a result of communications difficulties.

- Change management

The management of change is a vital part of any project. As mentioned in the discussion of user related challenges, user resistance occurs on some projects. Despite this, change management is often neglected: *"Yes, there are always problem areas. First of all in all the projects change management is often neglected".* (Case 6). This statement is supported by Case 4 (translated) where the interviewee mentioned the following: *"Change management, the users normally don't think of it, but it is a big problem".*

- Client culture

Software houses sometimes encounter problems when they have to implement a new system in an organisation where the culture is not ready for this change. For example, Case 6 noted the following: *"A culture that I believe was not ready to go onto an ERP system on a platform like this".* In instances like this, change management becomes extremely important. However, as mentioned in the discussion above, change management is often neglected.

This is supported by the notion that the organization's maturity and readiness for these changes can play a vital role in the success of an outsourcing relationship (Lemmon, 2005).

5.4.3 Project management

The discussion of the project management in software houses will be organised as follows: First the author will describe the project management style, followed by a discussion of the tools used during project management.

- Project management style

While studying the project management style followed in the different projects, it became apparent that the size of the projects has an influence on the project management style. In the very big project (Case 4) and the average size project (Case 6) the management style was found to be stricter than with the smaller projects. Furthermore, the role of the project manager was emphasized more. Case 6 stated, *"... with a project like this you need a very tight and strict project manager".* This view was also expressed in Case 4 (translated): "*Our strategy to sort the problems out was myself. I got more involved and tried to sort it out on a seniority level, through relationships and a top-down approach".*

In some of the smaller projects, as in case 2, there was a loose and open management style with a focus on the people: *"... we used open communication...You are able to keep your people very motivated because the project is structured around them...".* Furthermore, the smaller projects (cases 1, 2 and 5) tend to rely on the process of the systems development methodology to handle the challenges that they face. For example, Case 1 and Case 5 state that by following the process of proper design and documentation one can manage user expectations effectively. This is illustrated in the following statement: *"...you don't want the customer to come back and say 'Wasn't it supposed to do this?' So you initially have a well documented design that states what it is supposed to do and get the customer to agree on it"* (Case 1). The importance of documentation is also emphasized in Case 5: *"...sign off the whole project and a lot of documentation... and they know this is what we are going to do"* Case 2 uses a methodology that can adapt to the changing user requirements: *"You know, it probably comes down to the topic, just having a methodology that is adaptable to user changes"*

- The use of tools

Many project management tools are available to facilitate the process. In this study, five of the cases (1, 2, 4, 5, and 6) used tools to facilitate project management. Microsoft Project was used in three of the cases namely Cases 4, 5, and 6, Microsoft Excel in Case 2, and Mantis backtracking software was used on the very small project (Case 1).

Case 3 emphasised the use of a project plan, but did not indicate what tools they used to create the plan.

5.4.4 Client Relationship management in software houses

The management of the relationship between the client and the software house is an important responsibility. It has to be done throughout the project and plays an import role even after implementation was done successfully. In this study it became clear that people with a business background (or business knowledge) are responsible for the client relationship management in software houses.

According to Cases 4, 5 and 6, the responsibility of managing this relationship generally fell on people with a business related background. Case 4 (translated) stated specifically that client relationship management is the responsibility of *"... business people instead of IT people"*. Case 5 mentioned that the clients deal directly with the people from operations: *"...operations deals directly with the client .... so a client will contact them"* . These are the people on the project with a business background and knowledge of the business model. In Case 6 the interviewee mentioned that a person, called the account manager, would manage the relationship between the client and the software house. This account manager has a management background and knowledge of the specific business solution that they are working with. This is important, because *"they need to speak the same language as the client"*.

Case 3 contradicted the notion that relationship management should be done by a person with a business background. Case 3 (Translated) used an experienced developer to manage the relationship with the client: *"... a K-4 programmer would act as a team leader and be the one who talks to the users"*. After further analysis of why this contradiction occurred it was found that a K4- programmer would be a programmer or consultant with many successful implementations under his belt, and therefore is someone who also understands the business: *"The K4 would have some business knowledge as well, and would understand what the user meant from a business perspective*" (Case 3 (Translated)).

5.4.5 The perceptions of project team member in software houses regarding outsourcing

In this section the general perceptions of the project team members in software houses regarding outsourcing will be reported. This will be followed by a discussion of the perceived advantages and disadvantages of outsourcing as perceived by the project team members.

- General perceptions of project team members in software houses regarding outsourcing

All the cases agreed that it could be beneficial for companies to outsource their system development. This finding could be expected, because of the software houses' direct involvement (and benefit) in outsourcing. However, it was noted that outsourcing could be especially valuable for smaller companies. Case 3 (translated) argued, in support of Alner (2001), that the cost of keeping good developers at a small company exceeds the price paid to get a system developed by a software house: *"Your costs when doing in-house development and keeping the developers in house will not be as cost effective as outsourcing it, especially for the smaller companies"*. According to Case 4 (translated) it takes a lot of money to successfully develop systems in-house, and small companies do not always have the financial strenght to afford it: *"Your smaller guys that don't have many millions of rands like the banks and the big corporates. For the smaller guys it is definitely better to outsource"*. It was also mentioned by Case 6 that the development and management of enterprise resource planning systems for smaller companies should rather be outsourced: *"I think the management of the SAP application should be outsourced in a SME"*.

5.4.5.1 Perceptions regarding the advantages of outsourcing

In this section the perceived advantages of outsourcing from the project team members' point of view will be presented.

- Bigger knowledge base

Three cases (Case 3, Case 4 and Case 5) mentioned that through the experience obtained by working on many projects, a software house obtains a bigger knowledge base. This knowledge base allows the software house to provide the client with more than one possible solution and help to manage the changes needed in the company. The following was noted by Case 3 (translated): *"People that supply the outsourcing service get in contact with so many other clients and so many other industries and scenarios. They have the experience to manage the changes for a company and can possibly give better options to what the client should and shouldn't do".* This view was also expressed by Case 4 (translated): *"If you outsource, you have people who have worked on different business solutions, so they can solve the same problem in ten different ways"*, and by Case 5: *"...many people who know systems and possible solutions to problems".*

All the above mentioned statements on the bigger knowledge base agree with Alner (2001) who argues that, when client organisations outsource their information system development efforts, the client has access to the knowledge of high-level specialists who may not belong to his own staff.

- Personnel turnover

It is known that employees in the information technology field tend to change companies often. Case 3 (translated) stated the following: *"It is difficult to keep people that are really good. They would want to work on the big projects the whole time and will move around instead of staying with one company".* This presents the problem of trying to prevent your good developers from moving to another company. The following statement illustrates this: *"It is quite difficult for corporates to keep strong development teams"* (Case 2). At software houses the employees work on many projects and constantly get exposed to new technologies and new challenges. Therefore a software house is in a better position to keep the good developers and mangers as their employees: *"If you look today at IT people, they come and go and there is constantly new technology. You can't keep an in-house team happy and get the best quality out of them.... A definite advantage of outsourcing is that you don't have an issue with staff turnover"* (Case 4 Translated).

These statements confirm the idea that retaining top personnel, as discussed by Alner (2001), becomes the responsibility of the service provider, who should keep his personnel up to date and handle any turnover that might occur.

- Productivity improves

Another advantage of outsourcing system development is that the software houses tend to be more productive than an in house team. Case 2 mentioned that the driving force of a software houses' productivity is the fact that system development is their core business and that teams will work overtime to meet deadlines: *"...being our bread and butter, people who work for us understand that. And deadlines are often not negotiable and guys will burn midnight oil to do that"* (Case 2.) Case 3 (translated) stated that their productivity improved because of the way they push to make deadlines. *"Because we really push to make the agreed deadlines it also improves out productivity"* (Case 3).

According to Lemmon (2005), the productivity levels may vary throughout the life cycle of the outsourced project, but improved productivity is one of the reasons why companies outsource their system development.

- Developers are of a higher quality

According to Case 1 it is a lot easier to get developers, and get systems developed when you outsource: *"From a company's point of view it's very easy to get developers and systems developed when you outsource it".* Apart from that, Case 3 mentioned that the developers who work for a software house become better developers because they are constantly developing systems and learning from other people and projects: *" The people that supply the outsourcing service, because they are constantly out there and working on different project, and with different clients, they are probably your better quality developers".*

It is also mentioned by Alner (2001) as well as Gonzalez *et al.* (2005) that retaining top personnel is the responsibility of the service provider. The software house has to keep its personnel up to date with the latest technology and trends.

5.4.5.2 Perceptions regarding the disadvantages of outsourcing

During the study it was found that there were fewer disadvantages than advantages mentioned by the software houses. The disadvantages that stood out will be discussed next.

- The risk of choosing the wrong software house

It is extremely important to choose the right software house when outsourcing system development and even more so when outsourcing your entire infrastructure. Case 2 stated, *"choosing the wrong service provider would be a big risk"*. This was confirmed by Case 4 (translated): *"The only disadvantage that I can think of now is if you make a poor selection on who your service provider will be"*. Choosing the wrong service provider can be very costly to a company. The client company and the software house should be compatible and the software house should know the client's business in order to suggest the best possible solutions.

The disadvantage mentioned above can be managed as Avison and Fitzgerald (2003) mention, viz. that client companies who wish to outsource their information system development has to develop skills in selecting the correct vendor.

- Loss of intellectual property.

When system development is outsourced, the software house is obtaining information on your market place, the way you do business and information on how the system really works. Case 3 (translated) mentioned that: *"Many companies would want to keep their knowledge in-house"*. Case 2 expressed the following concern: *"IP (intellectual property) retention, where you are putting a company in a position to gain a lot of experience in your market place and you are not keeping that IP internally. So that's probably a big risk"*. As soon as that system is developed successfully the team who worked on the project takes all that newly obtained knowledge with them when they leave. Only a small amount of that knowledge is transferred to the client company through training and documentation.

These findings are supported by Avison and Fitzgerald (2003) who state that when organisations outsource their information systems development, the skills, experience and

expertise that the client organisation would have had if the systems had been developed in-house, are lost.

- Isolation between client and service provider

Case 1 mentioned a disadvantage of outsourcing, which is especially associated with offshore outsourcing. This is the distance between the clients and providers when they are not even on the same continent anymore: *"...there is such isolation between the development and the customers..."*. This creates communication problems and a challenging environment for relationship management.

This perception is supported by Weakland (2005) where the lack of proximity to staff is one of the concern factors regarding the outsourcing of information systems development.

5.5 Systems development methodology

In this section the deployment of systems development methodologies at project level in the software houses will be discussed. This discussion will focus on the use of methodologies, the perceived support provided by the methodologies and their impact on the development process and the system being developed.

5.5.1 The use of a systems development methodology

In this section the type of methodologies used as well as how strictly these methodologies are used will be discussed.

- Which methodologies were used

In all the projects system development methodologies were used. The ASAP methodology was used by four of the projects (Cases 3, 4, 5, 6), while Case 2 used a combination of XP and Agile methods. Case 1 used an in house methodology based on object-oriented design, which he adapted to fit his needs and style.

Four of the six projects studied used the ASAP methodology, which is part of the SAP enterprise resource planning system. According to Avison and Fitzgerald (2006), enterprise resource planning systems are realisations of method engineering goals. They provide a toolkit of integrated method fragments and tools for modelling and building applications. They also mentioned that enterprise resource planning systems are not just a set of methods and tools, but are also regarded as infrastructures that incorporate a type of front-end, back-end, and middle-tier architecture that runs on different networks and platforms (Avison and Fitzgerald, 2006). The market leader in supplying enterprise resource planning systems, SAP, has a 40 percent market share of the overall market (Woodie, 2005). This supports the finding that 4 of the 6 projects studied used the ASAP methodology during the development, customisation and implementation of a SAP – enterprise resource planning system. The following definition of the ASAP methodology was found on the SearchSap.com website:

> "SAP (AcceleratedSAP) is a step-by-step methodology for speeding up the implementation of an SAP/R3 system. The components of ASAP, which can be used together or individually, are called accelerators. Accelerators are based on the best practices of SAP customers from around the world and consist of a number of templates, questions, and scenarios that require user input to help the user determine the best way to implement their R/3 system. According to SAP literature, ASAP can reduce the time required to implement an R/3 system by as much as 50%. ASAP is shipped automatically with every R/3 delivery and upgrade. There is no charge for ASAP."- (SearchSap.com)

This finding supports the discussion in chapter three on the era of reassessment where Avison and Fitzgerald (2006) predict a movement to external development in a number of ways, including the use of packages and outsourcing. According to them, the package market is becoming increasingly sophisticated and client companies can buy a package that covers all their standard requirements. As seen in the finding above and in Avison and Fitzgerald (2006), the enterprise resource planning systems have emerged in the last few years and have become very popular. These packages are then tailored to fit the client organization's needs.

- How strict were the system development methodologies used

Not one project followed a methodology very strictly. All the projects seemed to agree that a methodology had to be used in a flexible enough manner to handle the changes and challenges that arise during the project. To give one example, Case 2 stated it precisely. *"We loosely base our development on the extreme development methodology, but you know only loosely, we don't follow it perfectly."* The other cases, although not stating it directly, gave the impression that they did not follow their methodologies very strictly, but rather used parts of it and changed it, basically using it as a guideline to direct the process.

This corresponds well with the contingency theory by Fitzgerald *et al.* (2002), that system development methodologies are very seldom used just as they are. Instead they are adapted and changed within an organisation.

5.5.2 Support provided by the system development methodology

The following is a discussion on the perceptions that the project team members in South African software houses have regarding the support that the systems development methodologies provide. The term support includes production support, support in control and also cognitive and cooperative support.

5.5.2.1 Control support

Support given by system development methodologies as a control technology includes project management support as well as ways of tracking progress and keeping to a schedule and budget.

- Structure and guidance

According to De Vries (2004), system development methodologies are used by software houses to structure the information systems development process. This is supported by the findings of this study as that in all of the cases mentioned, a methodology gives structure and guidance to the project, for example: *"... a good structure knowing where you're going..."* (Case 1). This view is also echoed by Case 3 (translated): *"...phases and deadlines that give direction..."* and Case 4 (translated): *"It gives structure and definitive*

guidelines". According to Case 2, following a system development methodology *"is a logical way of doing things".* In system development it is important to keep focus and have a plan that you as project team can follow to get the system developed successfully. This is illustrated by the following statement: *"... it's the most productive way to work, to work with a plan"* (Case 5). In Case 6 this plan is called a roadmap: *"The blueprint will then actually be our roadmap afterwards".* These roadmaps and plans all guide the project team members through the system development process.

The structure and guidance that the use of a systems development methodology gives are also discussed by Fitzgerald *et al.* (2002). He discusses the roles of methodologies in terms of rational and political categories. In both these categories the methodology gives direction to the process and reassurance to the people involved. In Avison and Fitzgerald (2006) and De Vries (2004), an important argument for the use of system development methodologies is presented. This included the notion that methodologies provide a standard process to follow and that the systematic activities specified by methodologies help organisations to keep track of system changes.

- Accountability

It was mentioned by both the non-SAP projects (Cases 1 and 2) that the methodology improves accountability. It almost provides a certain type of failsafe so that blame cannot fall on them. Case 1 mentioned the following regarding the advantages obtained by using a methodology: *"First of all it's accountability with regard to what has to be done and what's supposed to get done".* This view was also expressed by Case 2: *"The guys need to be accountable...".* The way the interviewees discussed accountability indicated that the use of the methodology and the documentation that it promotes, improve the accountability of the developers.

Although none of the SAP projects directly stated that accountability is improved by the use of the methodology, it was derived by the author that by all the regular sign off's, as stated in Cases 3, 4, 5 and 6, they also used the methodology to improve accountability for their actions.

5.5.2.2 Cognitive and cooperative support

Cognitive and cooperative support illustrate the way the methodology guides the way team members work together, and the influence it has on communication and the exchange of information.

- Communication

Improved communication through the use of a system development methodology is stated by Carayannis and Coleman (2005) and was mentioned as an advantage at all the cases in this study. There are various types of communication when developing a system in a software house. Both communication between team members and communication between the software house and the client were mentioned.

Most of these quotes indicate that communication with the client was improved by means of proper design and documentation which were guided by the methodology they used: *"So you initially have a well documented design that states what it's supposed to do and get the customer to agree on it"* (Case 1). Case 5 mentioned that when documents have to be signed before the next phase can be started, it helps the communication, especially between the client and the software house: *"...we have to provide the client with statements of work and get the clients to sign it off"*. The author derived from the interviews that the use of a methodology implicates the documentation of certain agreements between the software house and the client; these documents are seen as important communication enhancers. Keeping everyone informed on what the current status is certainly benefits both parties in the long run: *"I think the no surprises viewpoint is the big thing"* (Case 2). It is vital to communicate the changes regarding the old and new systems to the client, making sure that they grasp the impact that the new system will have on the way they used to work:*"... the blueprint stage we will give him (the customer) guidelines of what the changes will be and what the expected results of those changes will be... it is very important to make sure you understand and then create a document that states that he also likes what you are proposing"*. In a discussion regarding the advantages of using a systems development methodology Case 6 mentioned that it improves communication. *"...they (the team members and clients) communicate, it is what we have seen."* Case 6.

Case 4 mentioned that a team who knows the same methodology, already has a communication advantage above a team with members who used to use different or no methodologies. *"... You don't waste as much time round discussions on what other ways people have learned, you know everyone' s reasoning process is the same"* (Case 4 (Translated)).

- Better documentation

Better or guided documentation is another advantage of using a systems development methodology (Villarroela, *et al.* 2004). This advantage was mentioned by all of the cases. According to Case 6, *"...documentation in the SAP environment is essential".* Documentation is especially important for future upgrades (Case 1), and for the communication benefit when working with a dynamic team (Case 3 (Translated)). Furthermore, Case 3 (Translated) argued that documentation is vital although it takes up some time. Even comments in the code need to be put in place as the system are developed: *"Documentation is important, it takes a lot of time unfortunately".* Case 4 (translated) stated clearly that the methodology improves the quality of documentation. *"It certainly improves documentation and quality..... But it definitely helps with the quality, structure and documentation".*

Interestingly it was found that that all the cases agreed that an advantage of using a methodology is the way it improves documentation. In contradiction with the norm the developers were surprisingly in support of documentation. Only one case even mentioned that it takes a long time to do documentation (although still vital). In contrast to developers, software houses are very positive regarding documentation. It can be that software houses have a greater need for proper documentation because of the following reasons:

Client Relationship: It is very important to get everything on paper, what needs to be done, what has been done, and then get all of it signed off by the client.

Knowledge Management: In a software house, system development is the core business. The documentation of successfully completed projects can help solve a problem in the development of similar systems. The documentation is a way the software house keeps the

knowledge obtained through previous projects, and does not lose it when a developer or project manager leaves the company.

5.5.2.3 Production support

Support as production technology is where the use of a methodology provides the project team with tools, techniques and methods to help them with the development.

- Requirements analysis

The general feeling that surfaced was that requirement analysis is a difficult but a very important part of the system development process. In this study it was found that many of the cases used methodology based tools to improve the requirement analysis process.

- Use of tools

In response to the question "How do you determine what the users want?" Cases 3, 4 and 6 answered that the methodology includes the use of tools to help identify the user requirements.

Blueprints, solution composer, workshops and RACI matrix were all used when doing requirement analysis. Case 3 (translated) mentioned that by using the blueprint, which is designed in the blueprint phase, it is easier to communicate the changes regarding the old and new system to the client. *"..in the blueprint stage we will give him (the customer) guidelines of what the changes will be and what the expected impact of those changes will be"* Case 4 used a solution composer to define the scope and to generate a draft timeline for the project plan: *"Solution composer that we can use in certain situations to pin down the scope"*.

Case 6 mentioned the use of workshops to make sure that the team understands exactly what the users want, and they also use matrices to indicate who is responsible for what: *"...you workshop it with your client, by investing some time at the beginning the client can actually say and explain what they want to achieve. We listen to the client, then go back and say what we can offer to meet the client requirements. And that's why we say every*

single SLA is custom specific. Also we draw up a RACI matrix so that it is clear who has what roles and responsibilities" (Case 6).

5.5.3 Impact

5.5.3.1 The impact that the use of a system development methodology has on the system

Most of the cases (1, 3, 4 and 6) showed that the use of the methodology had a positive impact on the system. These impacts include the following:

- Functionality

According to Case 3, functionality is improved because the development process, of the methodology, includes thorough planning and testing: *"In a positive way the functionality, it gets properly tested and because there are deadlines and because it is thoroughly thought through in the blueprint and planning".*

The improvement in the functionality of the system is also mentioned by Avison and Fitzgerald (2006). They state that the use of a methodology puts emphasis on meeting the user requirements.

- Quality

Cases 1, 3 and 4 supported Huisman & Iivari (2002b) when they stated that the use of a system development methodology improves the quality of the system. Case 1 stated: *"Quality is a great improvement" and* Case 4 (translated) mentioned: *"It definitely improves the documentation and the quality...".* Case 3 (translated) mentioned that the reason for this improvement in quality is mainly that the use of the system development methodology facilitates, and allows for, extra time to be spent on the planning and design phases: *"Quality, because you test and spend a lot of time on planning"*.

This finding supports one of the arguments for the use of methodologies that has been discussed in chapter 3. It is said that the use of a systems development methodology improves the quality of the outcome. According to Avison and Fitzgerald (2006), this is

done by providing a framework of processes with measurement and criteria for their execution.

As mentioned above, three of the cases (1, 3 and 4) mentioned that the use of a system development methodology improves the quality of the system. This was contradicted by one case, Case 2, who made the statement that quality was not determined by the methodology but by the people working on the project: *"Quality, we certainly believe that quality is based on the individual that you put on the project. So if you put an individual on a project that writes good quality code, you know in a broadly structured way then your end product will be of a high quality too".*

Case 2 agrees with Fitzgerald *et al.* (2002), by stating that the developer is more important than the methodology. It might be that this software house is sceptical of giving the methodology the credit as this could mean any group of people (in-house or not) can achieve the same outcome. There is an ongoing debate on this issue. Fitzgerald *et al.* (2002) argue that the success of a system development project is the direct result of the people working on the development thereof, and has very little to do with the methodology that they follow.

- Upgradeability / Expandability / Maintainability

Findings by Villarroela *et al.*(2004) as well as Avison and Fitzgerald (2006) are supported by the statements found in Cases 1 and 6, when they stated that the use of a system development methodology has an undeniable influence on the long term upgradeability, expandability and maintainability of a system: "*But in the long term the company might want the system upgraded and if it isn't documented it will be very difficult. That's another important factor of the process, the final software gets documented... and you have a system that's well designed, object oriented, highly expandable and maintainable"* (Case 1).

Case 6 mentioned that following the ASAP methodology helps them develop a system that is on standard and therefore is easier to maintain and improve: *"So when it comes to an upgrade that you can actually do a technical upgrade and don't have to spend money for a reimplementation because you deviated from standard".*

Case 6 agrees with Avison and Fitzgerald (2006) by saying that standardization is a very important advantage of using a methodology, especially with support in mind: *"And SAP also actually stresses that, because if you deviate from standard they do not support you".*

5.5.3.2 The impact that the use of a system development methodology has on the process

There was an even more positive result when looking at the impact the use of a system development methodology has on the development process. As mentioned earlier, five of the cases agreed that a methodology gives guidance to the project.

- Finish on time

In addition to guidance, the author found that three of the six cases (2, 3 and 6) mentioned that the use of the systems development methodology helps the team to finish the project on time: *"We did not have one delay"* (Case 6). Case 2 mentioned that by using a methodology the system is developed faster: *"In terms of the Project? Speed- Faster and Cost cheaper".* By having a project plan and meeting the deadlines the chances are very good of completing the project on time: *"I think in a positive way, if you say performance, definitely, people try to stick to deadlines and there is very good progress on such a project. The people normally do their best to stick to the plan"* (Case 3 (Translated)).

This finding supports Avison and Fitzgerald (2006) who state that the use of a system development methodology accelerates the development process.

- Improved productivity

The use of system development methodologies generally results in the improvement in the productivity of the team. This topic has been discussed by many authors including Huisman and Iivari (2002a) (2002b), Howard *et al.* (1999) and Avison and Fitzgerald (2006). In this study it was again mentioned by two cases (3 and 5) that using the methodology improved their productivity: *"Because we constantly push to make deadlines, our productivity is also good"* (Case 3 Translated). Case 5 (translated) mentioned that by

working under a guided process towards a deadline, production is also improved: *"Because we push to make the deadlines, the productivity also improves".*

5.6 Summary

In this chapter the author presented the findings of this study. These findings have been categorised into the following sections: environment related findings, which include user and environment related challenges, project management, client relationship management and perceptions of project team members regarding outsourcing. After the environment related findings were given, the author discussed the findings in terms of the use of system development methodologies. These were presented in subcategories such as use, support and impact.

Chapter 6
Summary and final conclusions

6.1 Introduction

This study extends previous research of Huisman and Iivari (2002a, 2002b) who studied the deployment of system development methodologies at organisational level and at individual level. The focus of this study was on the deployment of systems development methodologies at project level in software houses in South Africa. In this chapter the findings of the study will be summarized and limitations of this study as well as future work will be presented.

6.2 Research contributions

Although research has been done on systems development methodologies, there is generally a lack of research that focuses on the deployment of systems development methodologies at project level. Even more so is lack of research on software houses. Software houses are increasingly important in the information system development field, especially when taking into account the movement to external development and outsourcing.

The general aim of this study was to study the deployment of systems development methodologies at project level in software houses in South Africa.
More specifically, the author aimed to understand and explain the following:

- The context and the outsourcing environment in which system development and the possible use of systems development methodologies take place.
- The use of systems development methodologies at project level in software houses in South Africa.
- The perceived support that the systems development methodologies provide. The term support includes production support, support in control and also cognitive and cooperative support.

- The perceived impact of systems development methodologies on the developed system and the development process.

In this study the term systems development methodology is seen as a combination of the following:

- A systems development approach/approaches:
  The philosophical view on which the methodology is built. It entails the set of goals, guiding principles and beliefs, fundamental concepts and principles of the systems development process that drive interpretations and actions in the development of systems. Examples are the structured approach, object oriented approach, information modelling, etc.
- A systems development process model/models:
  A representation of the sequences of stages through which a system evolves. Some examples are the linear life cycle model and the spiral model.
- A systems development method/methods:
  A systematic approach to conducting at least one complete phase of systems development, consisting of a set of guidelines, activities, techniques and tools, based on a particular philosophy of systems development and the target system. Examples include OMT, IE, etc.
- A systems development technique/techniques:
  A procedure, possibly with a prescribed notation, to perform a development activity, for example entity relationship diagrams.

The term deployment includes the following three aspects:

- The use of systems development methodologies at project level in software houses in South Africa.
- The perceived support that the systems development methodologies provide. The term support includes production support, support in control and also cognitive and cooperative support.
- The perceived impact of systems development methodologies on the developed system and the development process.

6.2.1 The perceptions of project team members in software-houses regarding the environment that software houses operate in

In this study the environment in which software houses operate was studied. Certain recurrent themes became apparent in the material gathered from each case.

- Challenges

The findings showed that there are many challenges in the development environment. These challenges can be categorised into two groups, namely user related challenges and environment related challenges.

The user related challenges consisted of: the constant changing of user requirements, the difficulty of managing client expectations, the availability of the users, the resistance to change posed by some user. Computer literacy and expertise was another concern followed by challenges regarding the communication and generation gaps between the development team and the users. When categorising the cases into the different sizes of the projects, the author found a difference in the way the smaller projects handled the challenge of changing user requirements compared to larger projects. The big projects did not allow any changes while both small projects tried to accommodate the user by absorbing most of the changes.

The environment related challenges were found to be: the constantly changing environment, the difficulty of integrating with 3rd party systems, the neglecting of change management and the lack of readiness for change in the client organisations' culture.

- Project management

The project management style was found to be stricter on the bigger projects than on the smaller projects. Discussing the way problems were managed, it was found that the smaller projects tended to rely on the process of the systems development methodologies to handle their problem management, while the bigger projects relied more on management intervention. The use of tools to facilitate project management was found to be very popular.

- Client Relationship management in software houses

It was found that the management of the relationship between the client and the software house is an increasingly important responsibility. It has to be done throughout the project and plays an import role even after a successful implementation is done. Although the use of systems development methodologies was found to increase communication between the client company and the organisation, most of the cases stated that the main relationship management responsibility lies on the shoulders of team members who understand the clients business and has a business background.

- Perceptions of the project team members in software houses regarding outsourcing

All six the cases stated that it would be beneficial for companies to outsource their information systems development. They mentioned that outsourcing is especially valuable for smaller companies. The following is a list of perceived advantages of outsourcing that were found in this study: bigger knowledge base; personnel turnover; improves productivity; and the developers are of a higher quality.

The perceived disadvantages of outsourcing that were found included the risk of choosing the wrong service provider, the loss of intellectual property, and the isolation between the client and the service provider.

6.2.2 The use of a system development methodology

It was found that all six the projects used systems development methodologies. The four projects that involved enterprise resource planning systems all followed the ASAP methodology. The other two projects used in-house methodologies based on different parts of the object oriented as well as the extreme programming methodologies; they changed these parts according to their needs, and used the methodologies to guide their development processes.

The main idea in terms of methodology use that was derived from the interviews, was that not one project followed a methodology very strictly. All the projects seemed to agree that a methodology had to be used in a flexible manner in order to handle the changes and challenges that arise during the project.

6.2.3 The support provided by the use of a systems development methodology

- Control support

It was found that a systems development methodology gives structure and guidance to the project. Another finding is that the use of a systems development methodology improves accountability. Accountability is especially important when taking into account the outsourcing environment that software houses operate in.

- Cognitive and cooperative support

In this study it was found that the cognitive and cooperative support given by the use of a system development methodology can be obtained through improved communication and better documentation. All the cases agreed that the use of systems development methodologies resulted in better communication. This includes both communication between team members and communication between the software house and the client. Most of the interviewees indicated that communication was improved by means of proper design and documentation which were guided by the methodology they used. As mentioned above, documentation plays an important role in the cognitive and cooperative part of a project. It was found that better documentation is an important advantage of systems development methodologies.

- Production support

The general feeling that surfaced was that requirement analysis is a difficult but very important part of the system development process. In this study it was found that many of the cases used methodology based tools to improve the requirement analysis process. This indicates the use of systems development methodologies as a supporting production technology. The tools that were used included blueprints, a solution composer, workshops and RACI matrices. These tools where all used to improve and enhance the requirement analysis process.

6.2.4 The impact of the use of a systems development methodology

- Impact on the system

In this study most of the cases indicated that the use of a systems development methodology had a positive impact on the system. The following system improvements were found: functionality; quality; upgradeability; expandability; maintainability.

- The Impact on the process

It was found that the use of systems development methodologies had an even more positive result on the development process. As mentioned earlier, five of the cases agreed that a methodology gives guidance to the team during the development of the project. It was found that the use of a methodology resulted in projects that finished on time with improved productivity.

6.3 Limitations of this study and future work

In this study six cases were studied. Although six cases are enough for an interpretive study, they are not enough to build a theory regarding the deployment of systems development methodologies at project level in software houses in South Africa. The author would propose using more cases along with grounded theory in future work to aid in theory building on the subject.

A qualitative approach has been followed in this study. Doing a quantitative study to compare findings should be very interesting.

SAP is the growing market leader in the field of enterprise resource planning systems in South Africa. This was also reflected in this study. Perhaps identifying a more diverse set of projects could also improve the credibility of the study.

There has been some debate about both the conceptual and practical value of systems development methodologies for improving the way that systems are developed. The

findings from this study reflect this debate. The author proposes further research in order to understand why these different perceptions exist.

Finally, analysis of different perspectives on the use of systems development methodologies suggests that, while they may meet the needs of certain projects, it is necessary to deploy them in a flexible manner as to allow for changes. These changes include changing user requirements as well as changes in the environment that information systems are developed in. Agile systems development methodologies aim to address these changes in a speedy and flexible manner. One aspect of agile systems development methodologies is the fact that only "light" documentation is required. However, in this study we have found that documentation is very important during the development of projects in software houses, as it improves communication and accountability. In future work the suitability of agile systems development methodologies for use in software houses should be determined.

References

AGARWAL, R. 1996. Cognitive fit in requirements modeling: a study of object and process methodologies. Journal of management information systems, 13 (2):137–162.

AKOMODE, O.J., LEES, B. & INGENS, C.H. 1998. Constructing customized models and providing information to support IS outsourcing decisions. *Logistics information management,* 11(2):114-127.

ALNER, M. 2001. The effects of outsourcing on information security. *Information systems security,* 10(2):35-43.

ANNON. 2002 CHILD CARE & EARLY EDUCATION RESEARCH CONNECTIONS. Qualitative research assessment tool. http://www.childcareresearch.org/discover/datamethods/downloads/QualitativeResAssessTool.pdf Date of access: 13 Oct. 2006.

AUSTRADE. 2006. Information and communications technology to South Africa: trends and opportunities. http://www.austrade.gov.au/australia/layout/0,,0_S2-1_CLNTXID0019-2_-3_PWB110763335-4_-5_-6_-7_,00.html Date of access: 13 Oct. 2006.

AVISON, D. & FITZGERALD, G. 2002. Information systems development: methodologies, techniques and tools. 3rd ed. London: McGraw-Hill. 416p.

AVISON, D. & FITZGERALD, G. 2003. Where now for development methodologies. *Communications of the ACM,* 46(1): 78-82.

AVISON, D.E. & FITZGERALD, G. 2006. Information systems development, methodologies, techniques and tools. 4th ed. London: McGraw-Hill. 645p.

BAHLI, B. & RIVARD, S. 2005. Validating measures of information technology outsourcing risk factors. *OMEGA: the international journal of management Science,* 33(2): 175-187.

BALL, L. J. & ORMEROD, T. C. 2000. Putting ethnography to work: the case for a cognitive ethnography of design. *International journal human-computer studies.* 53 : 147-168. http://www.psych.lancs.ac.uk/people/uploads/TomOrmerod20060328T143129.pdf Date of access: 10 Oct. 2006 .

BARRY, C. A. 1998. Choosing qualitative data analysis software: Atlas/ti and Nudist compared. *Sociological research online,* 3(3). http://www.socresonline.org.uk/3/3/4.html Date of access: 12 Apr. 2006.

BARTHELEMY, J. 2001. The hidden cost of IT outsourcing. *MIT Sloan management review,* 42(3):60-69.

BRYMAN, A. 2004. Qualitative research on leadership: a critical but appreciative review. *The leadership quarterly.* 15(6): 729-769.

CARAYANNIS, E. & COLEMAN, J. 2005. Creative system design methodologies: the case of complex technical systems. *Technovision.* 25(8): 831-840.

CARVALHO, L., SCOTT, L. & JEFFERY, R. 2005. An exploratory study into the use of qualitative research methods in descriptive process modelling. *Information and software technology,* 47(2): 113-127.

CHOUDHURY, V. & SUBHERWAL, R. 2003. Portfolio of control in outsourced software development projects. *ISR,* 14(3):291-314.

CRESWELL, J. W. 2003. Research design: qualitative, quantitative, and mixed methods approaches. 2nd ed.. Thousand Oaks, Calif.: Sage. 246p.

DE MARCO, T. 1982. Controlling software projects: management measurement and estimation. Englewood Cliffs, Yourdon, NJ. 284p

DE VILLIERS, D. J. 2002. Introducing the RUP into an organization. *The rational edge e-zine for the rational community,* (1):48-62, Jan. http://download.builder.ibm.com/ibmdl/pub/software/dw/rationaledge/jan02/TheRationalEdge200201Issue.pdf Date of access: 26 Jun. 2006.

DE VRIES, M. 2004. Evaluating or selecting a suitable information systems development methodology: a case study. *SA journal of industrial engineering,* 15(2):9-25.

DUÉ, R.T. 1992. The real cost of outsourcing. *Information systems management,* 9(1):78-81.

EARL, M.J. 1996. The risk of outsourcing IT. *Sloan management review,* 37(3):26-32.

EISENHARDT, K. M. 1989. Building theories from case study research, *Academy of management review,* 14(4):532-550.

ENDACOTT, R. 2005. Clinical research 4: qualitative data collection and analysis. *Intensive and critical care nursing,* 21(4):123-127.

ERLANDSON, D.A., HARRIS, E.L., SKIPPER, B.L. & ALLEN, S.D. 1993. Doing naturalistic inquiry: a guide to methods. Newbury Park, Calif.: Sage. 198p.

FITZGERALD, B. 1996. Formalized systems development methodologies: a critical perspective. *Information systems journal,* 6(1):3-23.

FITZGERALD, B. 1998. An empirical investigation into the adoption of systems development methodologies. *Information and management,* 34(6):317-328.

FITZGERALD, B. 2000. Systems development methodologies: the problem of tenses. *Information technology and people,* 13(3):174-185.

FITZGERALD, B., RUSSO, N.L. & O'KANE, T. 2003 Software development method tailoring at Motorola. *Communications of the ACM.* 46(4): 65- 70.

FITZGERALD, B., RUSSO, N. L. & STOLTERMAN, E. 2002. Information systems development: methods in action. London: McGraw-Hill. 194p.

GLASER, B. G. & STRAUSS, A.L. 1967. The discovery of grounded theory. Hawthorne, N.Y.: Aldine de Gruyter. 217p.

GLASSER, P. 1998. The knowledge factor. *Chief information officer,* 12(6):108-118.

GOLAFSHANI, N. 2003. Understanding reliability and validity in qualitative research. *The qualitative report,* 8(4):597-607.

GONZALEZ, R. GAZCO, J. & LLOPIS, J. 2005. Information systems outsourcing reasons in the largest Spanish firms. *International journal of information management,* 25(2): 117-136.

GROVER, V., CHEON, M. J. & TENG, T. C. 1994. A descriptive study on the outsourcing of information systems functions. *Information and management,* 27(1):33-44.

GROVER, V., CHEON, M. J. & TENG, T. C. 1996. The effect of service quality and partnership on the outsourcing of information service functions. *Journal of management information systems,* 12(4):89-116.

GUPTA, G. & GUPTA, H. 1992. Outsourcing the IS function. Is it necessary for your organization? *Information systems management,* 9(3):44-50.

HAMEL, G. & PRAHALAD, C.K. 1994. Competing for the future, Boston. Mass.: Harvard Business School Press. 327p.

HANCOCK, B. 1998. Trent focus for research and development in primary health care: an introduction to qualitative research. http://faculty.uccb.ns.ca/pmacintyre/course_pages/MBA603/MBA603_files/IntroQualitativeResearch.pdf

HARDY, C.J., THOMPSON, J.B. & EDWARDS, H.M. 1995. The use, limitations and customization of structured systems development methods in the United Kingdom. *Information and software technology,* 37(9): 467-477. Date of access: 26 Jun. 2006.

HERRIOT, R. E. & FIRESTONE, W. A. 1983. Multisite qualitative policy research: optimizing description and generalizability. *Educational researcher,* 12(2): 14-19.

HOWARD, G.S., BODNOVICH, T., JANICKI, T., LIEGLE, J., KLEIN, S., ALBERT, P. & CANNON, D. 1999. The efficacy of matching information systems development methodologies with application characteristics – an empirical study. *The journal of systems and software,* 45(2):177-195.

HUISMAN, H.M. & IIVARI, J. 2002a. The organizational deployment of systems development methodologies. (In Kirikova, M., Grundspenski, J., Wojtkowski, W., Wojtkowski, W.G., Wryza, S. & Zupanicic, J., eds. Information systems devekopment: advances in methodologies, components and management. New York: KLuwer Academic/ Plenum Publishers. P87-100.)

HUISMAN, H.M. & IIVARI, J. 2002b. The individual deployment of systems development methodologies, (In Pidduck, A.B., Mylopoulos, J. Woo,C.C. & Ozsu, M.T., eds. Advanced information systems engineering: 14th international conference CAISE 2002, Toronto, Canada, May 27-31, 2002. Lecture notes in computer science. Vol. 2348. p134-150.)

HUISMAN, M. 2000. The deployment of systems development methodologies: a South African experience. Potchefstroom: PU for CHE. (Thesis – PhD.) 242p.

HUISMAN, M. & IIVARI, J. 2006. Deployment of systems development methodologies: perceptual congruence between IS managers and systems developers. *Information & management,* 43(1): 29-49.

IIVARI, N. 2006. Representing the user in software development – a cultural analysis of usability work in the product development context. *Interacting with computers,* 18(4): 635-664.

JAYARATNA, N. 1994. Understanding and evaluating methodologies: NIMSAD. a systematic framework. London: McGraw-Hill. 259p.

JUDD, C.M., SMITH, E.R. & KIDDER, L.H. 1991. Research methods in social relations. 6th ed. Fort Worth: Holt, Rinehart and Winston. 573p.

KIMBALL, R., REEVES, L., ROSS, M. & THORNTHWAITE, W. 1998 The data warehouse lifecycle toolkit. New York: Wiley. 771p.

KING, W. R., & MALHOTRA, Y. 2000. Developing a framework for analyzing IS outsourcing. *Information and management,* 37(6):323-334.

KISHORE, R., RAO, H.R., NAM, K., RAJAGOPOLAN, S. & CHAUDHURY, A. 2003. A relationship perspective on IT outsourcing. *Communications of the ACM,* 46(12): 87-92.

KLEIN, H.K. & MYERS, M.D. 1999. A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS quarterly.* 23(1):67-94.

KRUCHTEN, P. 2001. What is the rational unified process?. *The rational edge e-zine for the rational community,* (1):4-14, Jan. http://download.boulder.ibm.com/ibmdl/pub/software/dw/rationaledge/jan01/TheRationalEdgeJan01Issue.pdf Date of access: 26 Jun. 2006.

LACITY, M.C. & HIRSCHHEIM, R. 1993. The information systems outsourcing bandwagon. *Sloan management review,* 35(1):73-86.

LACITY, M.C & WILLCOCKS, L.P. 1998. An empirical investigation of information technology sourcing practices: lessons from experience. *MIS quarterly*, 22(3):363-408.

LACITY, M.C., WILLCOCKS, L.P. & FEENY, D.F. 1996. The value of selective IT sourcing. *Sloan management review,* 37(3):13-15.

LANGLEY, A. 1999. Strategies for theorizing from process data. *Academy of management review,* 24(4):691-710.

LEHMANN, H. & GALLUPE, B. 2005. Information systems for multinational enterprises—some factors at work in their design and implementation. *Journal of international management,* 11(2)):163-186.

LEMMON, S. 2005. Outsourcing requires careful planning. http://networkworld.com/news/2005/072005-outsourcing.html Date of access: 21 Apr. 2006.

LINCOLN, Y.S. & GUBA, E.G. 1985. Naturalistic enquiry. Thousand Oaks, Calif: Sage Publications. 416p.

MACAULAY, C., BANYON, D. & CRERAR, A. 2000. Ethnography, theory and systems design: from intuition to insight. *International journal human-computer studies,* 53 : 35-60. http://www.idealibrary.com Date of access: 10 Oct. 2006.

MESO, P., MADEY, G., TROUTT, M. & LIEGLE, J. 2006. The knowledge management efficacy of matching information systems development methodologies with application characteristics – an experimental study. *The journal of systems and software,* 79(1):15-28.

NATIONAL FOREST FOUNDATION. 2006. Multiparty monitoring and assessment guidelines for community-based forest restoration in southwestern ponderosa pine forests. http://www.natlforests.org/pdf/monitoring_report.pdf Date of access: 12 Oct. 2006.

PALMQUIST, R.A. 1997. Uses and users of information. http://www.ischool.utexas.edu/~ssoy/usesusers/l391d1b.htm Date of access: 12 Oct. 2006.

PATTON, M.Q. 2002. Qualitative research and evaluation methods. 3rd ed. Thousand Oaks, Calif: Sage Publications. 598p.

POLKINGHORNE, D.E. 2005. Language and meaning: data collection in qualitative research. *Journal of counseling psychology,* 52(2):137-145.

PURVIS, R. & SAMBAMURTHY, V. 1997. An examination of designer and user perceptions of JAD and the traditional IS design methodology. *Information and management,* 32(3):123-135.

RAI, A. & AL-HINDI, H. 2000. The effects of development process modeling and task uncertainty on development quality performance. *Information and management,* 37(6):335-346.

ROGERS, E.M. 1995. Diffusion of innovations. 4th ed. New York: Free Press. 519p.

RUEGG, R. 2006. Bridging from project case study to portfolio analysis in a public R&D program: A framework for evaluation and introduction to a composite performance rating system. National Institute of Standards and Technology. Technology administration. U.S. Department of Commerce. http://www.atp.nist.gov/eao/gcr06-891/gcr06-891report.pdf. Date of access: 23 Oct. 2006.

SAEKI, M. 1998. A meta-model for method integration. *Information and software technology,* 39(14-15):925-932.

SANDERS, P. 1982. Phenomenology: a new way of viewing organizational research. Academy of management review, 7(3): 353-360.

SCHEIER, R. L. 1996. Outsourcing's fine print. *Computerworld,* 30(34):70.

SEALE, C., GOBO, G., GUBRIUM, J.F. & SILVERMAN, D. 2004. Qualitative Research Practice. Thousand Oaks, Calif.: SAGE. 620p.

SEAMAN, C. B. 1999. Qualitative methods in empirical studies of software engineering. *IEEE transactions on software engineering,* 25(4): 557-572.

SEARCHSAP.COM. 2006. Definition of ASAP. http://searchsap.techtarget.com/sDefinition/0,290660,sid21_gci871489,00.html. Date of access: 25 Oct. 2006.

SOY, S. K. 1996. *The case study as a research method.* http://www.gslis.utexas.edu/~ssoy/usesusers/l391d1b.htm. Date of access: 02 Jun. 2006.

SHAPIRO, S. 1997. Splitting the difference: the historical necessity of synthesis in software engeneering. IEEE annals of the history of computing, 19(1):20-54.

STRAUSS, A. & CORBIN, J. 1990. Basics of qualitative research: grounded theory procedures and techniques. Newbury Park, Calif.: Sage Publications. 270p.

UDO, G. 2000. Using analytic hierarchy process to analyze the information technology outsourcing decision. *Industrial management & data systems,* 100(9): 421-429.

URQUHART, C. 1997. Exploring analyst-client communication: using grounded theory techniques to investigate interaction in informal requirements gathering in information systems and qualitative research. (In Proceedings of the IFIP TC8 WG 8.2 international conference on information systems and qualitative research. London: Chapman & Hall. p149-181.)

VILLARROEL, R. FERNANDEZ- MEDINA, E. & PIATTINI,M. 2004. Secure information systems development: a survey and comparison. *Computers & security, 24(4):308-321.*

WALSHAM, G. 1993. Interpreting information systems in organisations. Chichester: Wiley. 269p

WEAKLAND, T. 2005. DiamondCluster 2005 global IT outsourcing study. http://exchange.diamondcluster.com/intelligence/200605/DiamondCluster2005OutsourcingStudy.pdf Date of access: 05 Sep. 2005

WEIDENBAUM, M. 2005. Outsourcing: pros and cons. *Business horizons.* 48(4): 311-315.

WOODIE, A. 2005. ERP market grew solidly in 2004, AMR research says. *The four hundred – iseries and AS/400 insight,* 14(25). http://www.itjungle.com/tfh/tfh062005-story03.html Date of access: 25 Oct. 2006.

WHITE. D. 2003. DRG outsourcing describes BPO... and the South African HRO industry climate, service level standards, and opportunities for international groups to enter the South African market. http://www.sovereign-publications.com/drgoutsourcing-ft.htm Date of access: 15 Oct. 2006.

WILLCOCKS, L. & FITZGERALD, G. 1994. Toward the residual IS organization? Research on IT outsourcing experiences in the United Kingdom. (In Baskerville, R., Smithson, S., Ngwenyama, O. & De Gross, J., eds. Transforming organizations with information technology. Amsterdam: North-Holland. p.129– 152.)

YIN, R. K. 1993. Applications of case study research. Newbury Park, Calif.: Sage Publications. 131p.

YIN, R. K. 2003. Case study research, design and methods. Thousand Oaks, Calif.: Sage Publications. 179p.